Optimization in Graphical Models

A survey with relations with LP

T. Schiex, MIAT, INRA

M. Cooper, S. de Givry, J. Larrosa, G. Verfaillie
Many more co-workers and contributors, see bibliography

September 2015 - JFRO - Paris

What will we see ?

On the Menu

1. Graphical models
   - Constraint programming
   - Cost Function Networks (Weighted CSP or WCP)
   - Stochastic Graphical Models
2. Solving techniques (tree search and *local consistency*)
   - Constraint programming
   - Stochastic Graphical Models
   - Cost Function Networks (WCSP/WCP)
   - Arc consistency, LP and duality
3. A quick list of solver techniques in toulbar2
4. A CFN application: Computational Protein Design

What is C(S)P ?

Solving a Constraint Network

1. Set $X \ni x_i$ of n variables, with finite domain D^i ($|D^i| \leq d$)
2. Set $C \ni c_S : D^S \rightarrow \{0,1\}$ of e constraints
3. c_S has scope $S \subset X$ ($|S| \leq r$)
4. Defines a **factorized** joint constraint over X:

$$\forall t \in D^X, C(t) = \max_{c_S \in C} c_S(t[S])$$

What is C(S)P ?

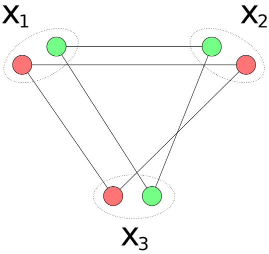
Solving a Constraint Network

1. Set $X \ni x_i$ of n variables, with finite domain D^i ($|D^i| \leq d$)
2. Set $C \ni c_S : D^S \to \{0, 1\}$ of e constraints
3. c_S has scope $S \subset X$ ($|S| \leq r$)
4. Defines a **factorized** joint constraint over X:

$$\forall t \in D^X, C(t) = \max_{c_S \in C} c_S(t[S])$$

Graph coloring/RLFAP-feas

1. A graph $G = (V, E)$ and m colors.
2. Can we color all vertices in such a way that no edge connects two vertices of the same color ?



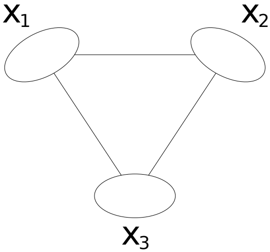
Constraint networks are graphical models

A CN defines...

A factorization (hyper)graph

1. Vertices as variables
2. Scopes/factors as (hyper)edges

Or bipartite incidence graph [KFL01]



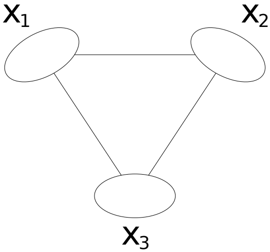
Constraint networks are graphical models

A CN defines...

A factorization (hyper)graph

1. Vertices as variables
2. Scopes/factors as (hyper)edges

Or bipartite incidence graph [KFL01]



Common to all factorization based models

- Factorization allows for conciseness
- Factorization usually leads to NP-hardness
- Factorization allows for "local" processing

Arbitrary concise constraints

- Table constraints (bounded scope)
- Short list of (non) solutions (arbitrary scopes, SAT clauses)

Arbitrary concise constraints

- Table constraints (bounded scope)
- Short list of (non) solutions (arbitrary scopes, SAT clauses)

Global constraints, arbitrary scope S

- AllDiff(S) (all different values)
- GCC($S, v_1, lb_1, ub_1 \ldots$). Satisfied iff the assignment of S contains between lb_i and ub_i occurrences of value v_i.
- Regular($\vec{S}, \mathcal{A}$): satisfied iff the tuple of values (word) defined by $\vec{S}$ is accepted by the finite state automata $\mathcal{A}$.
- See the catalog (http://sofdem.github.io/gccat)

Arbitrary concise constraints

- Table constraints (bounded scope)
- Short list of (non) solutions (arbitrary scopes, SAT clauses)

Global constraints, arbitrary scope S

- AllDiff(S) (all different values)
- GCC($S, v_1, lb_1, ub_1 \ldots$). Satisfied iff the assignment of S contains between lb_i and ub_i occurrences of value v_i.
- Regular($\vec{S}, \mathcal{A}$): satisfied iff the tuple of values (word) defined by $\vec{S}$ is accepted by the finite state automata $\mathcal{A}$.
- See the catalog (http://sofdem.github.io/gccat)

- not limited to linearity over integers (modeling)
- very long list of specific constraint types (modeling)

Many applications to real-world problems

Job-Shop scheduling

1. Set of tasks t_i with duration d_i and ressource (x_i starting time)
2. Precedence constraints $(t_i \rightarrow t_j)$ $(x_i + d_i \leq x_j)$
3. Deadline dd_i for final tasks t_i (no successors, $x_i + d_i \leq dd_i$)
4. non sharable ressources (t_i before or after t_j)

Many applications to real-world problems

Job-Shop scheduling

1. Set of tasks t_i with duration d_i and ressource (x_i starting time)
2. Precedence constraints $(t_i \rightarrow t_j)$ $(x_i + d_i \leq x_j)$
3. Deadline dd_i for final tasks t_i (no successors, $x_i + d_i \leq dd_i$)
4. non sharable ressources (t_i before or after t_j)

A variety of solvers: GeCode, Choco, AbsCons, JACOP, Mistral, MiniCSP, IBM Ilog, Cisco Eclipse. Many more global constraints for specific scheduling problems, cumulative resources and also for other domains.

See CP and CP/AI/OR application papers for more applications.

Job-Shop scheduling, min average tardiness

1. One extra "local cost" variable per final task: lateness l_i
2. A constraint to define it $l_i = \max(0, x_i + d_i - dd_i)$
3. A global cost variable x_{gc}
4. A global constraint to define it $x_{gc} = \sum l_i$

CSP/CP targeted at feasibility

Job-Shop scheduling, min average tardiness

1. One extra "local cost" variable per final task: lateness l_i
2. A constraint to define it $l_i = \max(0, x_i + d_i - dd_i)$
3. A global cost variable x_{gc}
4. A global constraint to define it $x_{gc} = \sum l_i$

- introduction of extra (non-decision) 'cost' variables
- defined by suitable constraints (soft globals, sum, reified)
- iterative feasibility problems solving

Cost Function Networks aka Weighted Constraint Networks

- Variables and domains as usual
- Cost functions $W \ni c_S : D^S \to \{0, \ldots, k\}$ (k finite or not)
- Cost combined by (bounded) addition [SFV95; CS04].

$$cost(t) = \min(\sum_{c_S \in C} c_S(t[S]), k)$$

A solution has cost $< k$. Optimal if it has minimum cost.

Lifting CP to optimization

Cost Function Networks aka Weighted Constraint Networks

- Variables and domains as usual
- Cost functions $W \ni c_S : D^S \to \{0, \ldots, k\}$ (k finite or not)
- Cost combined by (bounded) addition [SFV95; CS04].

$$cost(t) = \min\left(\sum_{c_S \in C} c_S(t[S]), k\right)$$

A solution has cost $< k$. Optimal if it has minimum cost.

Benefits

- Defines feasibility and cost homogeneously
- A constraint is a cost function with costs in $\{0, k\}$
- Tables, analytic ($x_1 \cdot x_3 + x_2$), globals (WeightedRegular,. . .)
- No 'non-decision' variables (unless you want them)

Markov Random Fields

- Random variables X with discrete domains
- joint non normalized probability distribution $p(X)$ defined as a product of positive real-valued functions:

$$p(X = t) \propto \prod_{c_S \in C} c_S(t[S])$$

Massively used in 2/3D Image Analysis, Statistical Physics, NLP...

Markov Random Fields

- Random variables X with discrete domains
- joint non normalized probability distribution $p(X)$ defined as a product of positive real-valued functions:

$$p(X = t) \propto \prod_{c_S \in C} c_S(t[S])$$

Massively used in 2/3D Image Analysis, Statistical Physics, NLP...

For optimization (MAP = Maximum a posteriori), MRF and CFN are essentially equivalent after a $-\log$ transform.

Bayesian networks

- Random variables X and domains
- joint normalized probability distribution $p(X)$ as a product of conditional probability tables defined on a DAG.

$$p(X = t) = \prod_{c_{x|Pa(x)} \in C} c_{x|Pa(x)}(t[x]|t[Pa(x)])$$

Massively used in Uncertain reasoning in AI, many applications, commercial solvers.

Bayesian networks

- Random variables X and domains
- joint normalized probability distribution $p(X)$ as a product of conditional probability tables defined on a DAG.

$$p(X = t) = \prod_{c_{x|Pa(x)} \in C} c_{x|Pa(x)}(t[x]|t[Pa(x)])$$

Massively used in Uncertain reasoning in AI, many applications, commercial solvers.

For optimization (MPE, maximum probability explanation), BN and CFN are essentially essentially equivalent after a $-\log$ transform.

CFN solvers directly useful for MAP/MRF and MPE/BN (and vice-versa).

CFN as ILP/QP

Binary CFN as 01LP (infinite k, finite costs)

The MRF/CFN local polytope [Sch76; Kos99; Wer07]

$$\text{Minimize} \sum_{i,a} c_i(a) \cdot x_{ia} + \sum_{\substack{c_{ij} \in C \\ a \in D^i, b \in D^j}} c_{ij}(a,b) \cdot y_{iajb} \quad \text{subject to}$$

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

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

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

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

$nd + e.d^r$ variables. $n + 2ed$ contraintes.

Only nd variables

$$\min \sum_{i,a} c_i(a).x_{ia} + \sum_{\substack{c_{ij} \in C \\ a \in D^i, b \in D^j}} c_{ij}(a,b) \cdot x_{ia} \cdot x_{jb} \quad \text{subject to}$$

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

01 binary CFN as 01QP (infinite k, finite costs)

Quadratic Pseudo Boolean optimization[BH02]

$$
\begin{aligned}
\min \sum_i \quad & c_i(1).x_{ia} + c_i(0).(1 - x_{ia}) + \\
\sum_{c_{ij} \in C} \quad & c_{ij}(1,1) \cdot x_{ia} \cdot x_{jb} + \\
& c_{ij}(0,1) \cdot (1 - x_{ia}) \cdot x_{jb} + \\
& c_{ij}(1,0) \cdot x_{ia} \cdot (1 - x_{jb}) + \\
& c_{ij}(0,0) \cdot (1 - x_{ia}) \cdot (1 - x_{jb})
\end{aligned}
$$

Posiform QPBO. Also covers Weighted Max2SAT (or Max-cut).

CFN can concisely express a variety of problems

Problem	#inst.	n	d	e	a
MRF (`uai`)	319				
Linkage	22	1289	7	2184	5
DBN	108	1094	2	22793	2
Grid	21	6400	2	19200	2
ImageAlignment	10	400	93	3563	2
ObjectDetection	37	60	21	1830	2
ProteinFolding	21	1972	503	8816	2
Segmentation	100	237	21	886	2
MRF/CVPR (`hdf5`)	1453				
ChineseChars	100	17856	2	553726	2
ColorSeg	3	414720	4	2069714	2
ColorSeg-4	9	86400	12	258600	2
ColorSeg-8	9	86400	12	430202	2
GeomSurf-3	300	1133	3	5039	3
GeomSurf-7	300	1133	7	5039	3
InPainting-4	2	14400	4	42960	2
InPainting-8	2	14400	4	71282	2
Matching	4	20	20	210	2
MatchingStereo	2	166222	20	497849	2
ObjectSeg	5	68160	8	203947	2
PhotoMontage	2	514080	7	1540689	2
SceneDecomp	715	208	8	769	2

Problem	#inst.	n	d	e	a
CFN (`wcsp`)	281				
Auction	170	246	2	11528	2
CELAR	16	458	44	2335	2
Pedigree	10	10017	28	18875	3
ProteinDesign	10	18	198	171	2
SPOT5	20	1057	4	21786	3
Warehouse	55	1100	300	101100	2
Max-CSP (`xcsp`)	503				
BlackHole	37	205	50	1651	2
Coloring	22	450	6	6164	2
Composed	80	83	10	785	2
EHI	200	315	7	4715	2
Geometric	100	50	20	605	2
Langford	4	33	29	517	2
QCP	60	264	9	2662	2

Problem	#inst.	n	d	e	a
WPMS (`wcnf`)	427				
MIPLib	12	24776	2	107956	93
MaxClique	62	3321	2	378247	2
Haplotyping	100	216117	2	1188223	580
PackupWeighted	99	25554	2	70677	177
PlanningWithPref	29	69409	2	771883	372
TimeTabling	25	903884	2	2912882	36
Upgradeability	100	18169	2	105097	77

Problem	#inst.	n	d	e	a
CP (`minizinc`)	35				
AMaze	6	1573	17	3173	4
FastFood	6(1)	2	5	3	2
Golomb	6(3)	44	163	717	3
OnCallRostering	5(3)	2205	89	4513	4
ParityLearning	7	759	20	1440	4
VehicleRoutingProb.	5	11531	100	22999	4

Tree Search & *Arc Consistency*

Depth First Search + Arc Consistency

1. Do we have a proof of infeasibility (AC) ?
2. If yes backtrack (back to previous state)
3. Else choose a non singleton variable x_i (vertical)
4. Split its domain in disjoint subsets (branching)
5. For each subset (horizontal)
    1. restrict x_i domain to this subset and recurse

CP solving technology

Depth First Search + Arc Consistency

1. Do we have a proof of infeasibility (AC) ?
2. If yes backtrack (back to previous state)
3. Else choose a non singleton variable x_i (vertical)
4. Split its domain in disjoint subsets (branching)
5. For each subset (horizontal)
    1. restrict x_i domain to this subset and recurse

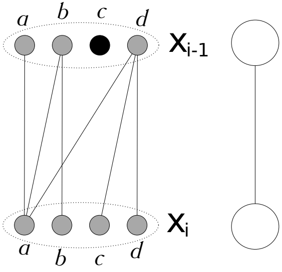
Why DFS ?

1. DFS is polynomial space
2. DFS benefits from AC incrementality for free
3. BFS would need even more memory for incrementality

Arc Consistency = local dynamic programming

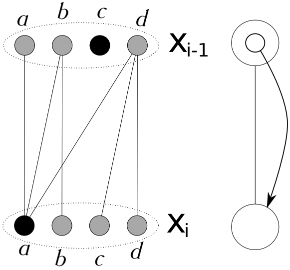
AC as Dynamic programming

- Imagine a CSP with linear graph
- Use DP to compute which values of x_i are part of a solution of $x_1, \dots x_i$ knowing those for x_{i-1}.



AC as Dynamic programming

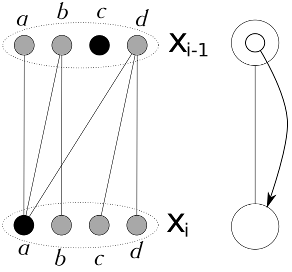
- Imagine a CSP with linear graph
- Use DP to compute which values of x_i are part of a solution of $x_1, \ldots x_i$ knowing those for x_{i-1}.



Arc Consistency = local dynamic programming

AC as Dynamic programming

- Imagine a CSP with linear graph
- Use DP to compute which values of x_i are part of a solution of $x_1, \dots x_i$ knowing those for x_{i-1}.



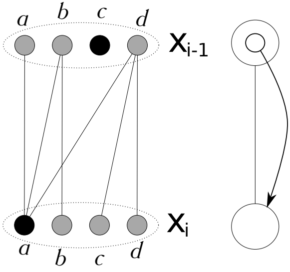
Revise = Equivalence Preserving Transformation (EPT)

- $a \in D^i$ cannot be part of a solution $(\nexists u \in D^j \mid c_{ij}(a, u) = 0)$.
- we can delete it.
- the resulting problem is equivalent (same set of solutions)
- internal incrementality (support)

Arc Consistency = local dynamic programming

AC as Dynamic programming

- Imagine a CSP with linear graph
- Use DP to compute which values of x_i are part of a solution of $x_1, \ldots x_i$ knowing those for x_{i-1}.



Revise = Equivalence Preserving Transformation (EPT)

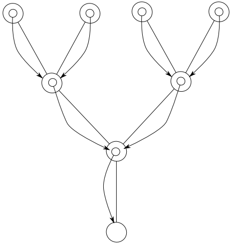
- $a \in D^i$ cannot be part of a solution ($\nexists u \in D^j \mid c_{ij}(a, u) = 0$).
- we can delete it.
- the resulting problem is equivalent (same set of solutions)
- internal incrementality (support)

Communication between constraints goes (only) through domains.

Directional AC, AC solve tree structured CN

Rooted tree CN

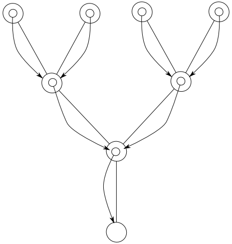
- Revise from leaves to root
- Root domain: only values part of a solution



Directional AC, AC solve tree structured CN

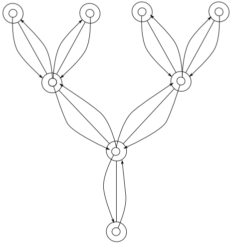
Rooted tree CN

- Revise from leaves to root
- Root domain: only values part of a solution



Tree CN

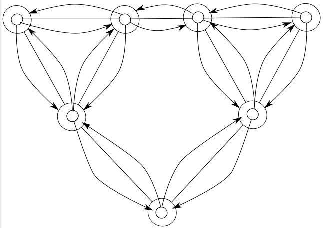
- Revise from leaves and back
- All domains: values part of a solution only
- Resulting problem solved backtrack-free [Fre82; Fre85]



Can be done on any CN, with arbitrary graph

Arc consistency

1. Linear time (tables)
2. Unique fixpoint (confluent)
3. Preserves equivalence
4. May detect infeasibility
5. Problem transformation (incremental)
6. internal incrementality (support)



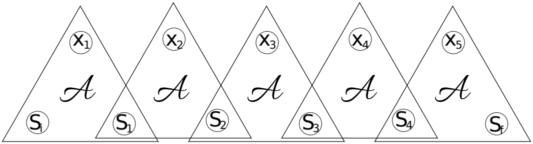
AC on global constraints

Global decomposable constraints [Bac07; QW06]

- Automata/CFG parsers, Knapsack: DP based.
- Enforcing AC on the global can be directly done by decomposing it in small constraints. Intermediary DP tables must be representable as "extra" variables in a tree CSP.
- Decomposable constraints emulate DP algorithms using AC.

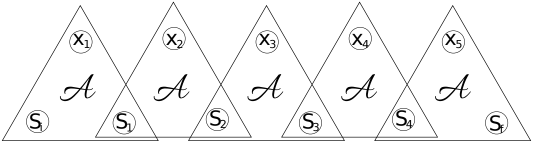
Global decomposable constraints [Bac07; QW06]

- Automata/CFG parsers, Knapsack: DP based.
- Enforcing AC on the global can be directly done by decomposing it in small constraints. Intermediary DP tables must be representable as "extra" variables in a tree CSP.
- Decomposable constraints emulate DP algorithms using AC.



Global decomposable constraints [Bac07; QW06]

- Automata/CFG parsers, Knapsack: DP based.
- Enforcing AC on the global can be directly done by decomposing it in small constraints. Intermediary DP tables must be representable as "extra" variables in a tree CSP.
- Decomposable constraints emulate DP algorithms using AC.

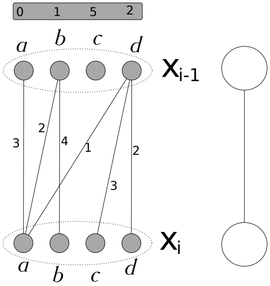


AllDiff (matching [Rég94]) not decomposable [Bes+09]. GCC (max flow [Rég96]), AllDiff with Cost variable (min-cost flow [VPR06])

Message passing in Markov Random Fields

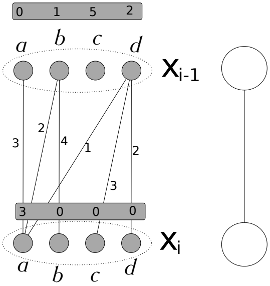
MP - Dynamic Programming [Pea88]

1. Use DP to compute the cost of an optimal solution that goes from x_1 to $a \in D_i$ knowing those for x_{i-1}
2. Use extra functions (messages) to store DP results



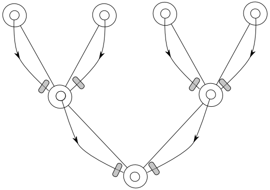
MP - Dynamic Programming [Pea88]

1. Use DP to compute the cost of an optimal solution that goes from x_1 to $a \in D_i$ knowing those for x_{i-1}
2. Use extra functions (messages) to store DP results



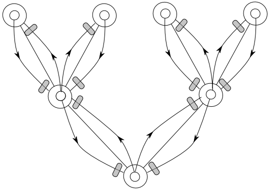
MP - Dynamic Programming [Pea88]

1. Use DP to compute the cost of an optimal solution that goes from x_1 to $a \in D_i$ knowing those for x_{i-1}
2. Use extra functions (messages) to store DP results



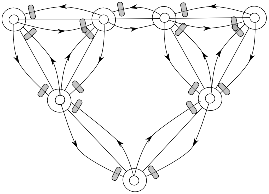
MP - Dynamic Programming [Pea88]

1. Use DP to compute the cost of an optimal solution that goes from x_1 to $a \in D_i$ knowing those for x_{i-1}
2. Use extra functions (messages) to store DP results



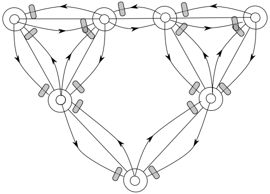
MP - Dynamic Programming [Pea88]

1. Use DP to compute the cost of an optimal solution that goes from x_1 to $a \in D_i$ knowing those for x_{i-1}
2. Use extra functions (messages) to store DP results



MP - Dynamic Programming [Pea88]

1. Use DP to compute the cost of an optimal solution that goes from x_1 to $a \in D_i$ knowing those for x_{i-1}
2. Use extra functions (messages) to store DP results

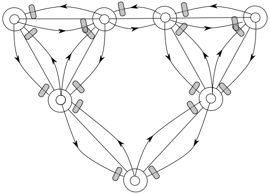


1. Solves Berge acyclic MRF/BN (acyclic Factor Graphs)
2. Does not converge on graphs (Loopy Belief Propagation)
3. Massively used to produce "good" solutions (turbo-decoding [RU01])
4. Not an equivalence preserving transformation [Pea88])

MRF/BN/Factor graphs ($-$ log domain)

MP - Dynamic Programming [Pea88]

1. Use DP to compute the cost of an optimal solution that goes from x_1 to $a \in D_i$ knowing those for x_{i-1}
2. Use extra functions (messages) to store DP results



1. Solves Berge acyclic MRF/BN (acyclic Factor Graphs)
2. Does not converge on graphs (Loopy Belief Propagation)
3. Massively used to produce "good" solutions (turbo-decoding [RU01])
4. Not an equivalence preserving transformation [Pea88])

Communication between functions goes through messages.

Solving the WCSP on a Cost Function Network

Depth First Branch and Bound + Arc Consistencies

1. Do we have a lower bound on optimum $\geq k$ ($c_\varnothing$)
2. If yes backtrack (back to previous state)
3. Else choose a non singleton variable x_i (vertical)
4. Split its domain in disjoint subsets (branching)
5. For each subset (horizontal)
    1. restrict x_i domain to this subset and recurse

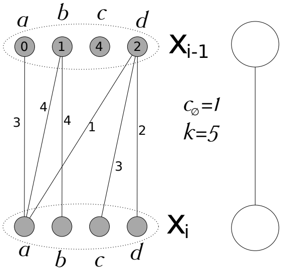
Depth First Branch and Bound + Arc Consistencies

1. Do we have a lower bound on optimum $\geq k$ ($c_\varnothing$)
2. If yes backtrack (back to previous state)
3. Else choose a non singleton variable x_i (vertical)
4. Split its domain in disjoint subsets (branching)
5. For each subset (horizontal)
   1. restrict x_i domain to this subset and recurse

1. When a solution is found, update k to its cost.
2. DFS vs. BFS: same arguments.

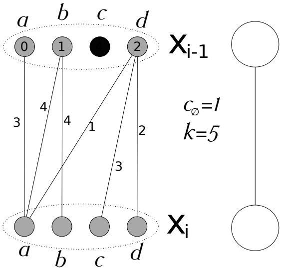
AC as Dynamic programming

1. MRF message passing but...
2. use $c_\varnothing$ and $c_i(a)$ to store optimum cost from x_1 to x_i
3. Preserves equivalence by "cost shifting" [Sch00; Sch76]



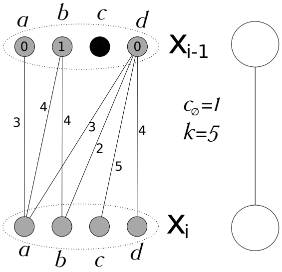
AC as Dynamic programming

1. MRF message passing but...
2. use $c_\varnothing$ and $c_i(a)$ to store optimum cost from x_1 to x_i
3. Preserves equivalence by "cost shifting" [Sch00; Sch76]



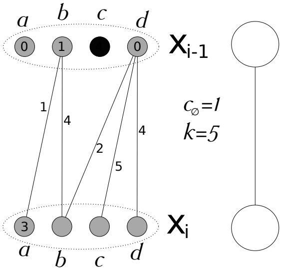
AC as Dynamic programming

1. MRF message passing but...
2. use $c_\varnothing$ and $c_i(a)$ to store optimum cost from x_1 to x_i
3. Preserves equivalence by "cost shifting" [Sch00; Sch76]



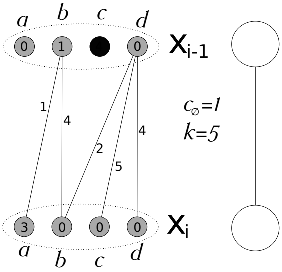
AC as Dynamic programming

1. MRF message passing but...
2. use $c_\varnothing$ and $c_i(a)$ to store optimum cost from x_1 to x_i
3. Preserves equivalence by "cost shifting" [Sch00; Sch76]



AC as Dynamic programming

1. MRF message passing but. . .
2. use $c_\varnothing$ and $c_i(a)$ to store optimum cost from x_1 to x_i
3. Preserves equivalence by "cost shifting" [Sch00; Sch76]



AC as Dynamic programming

1. MRF message passing but...
2. use $c_\varnothing$ and $c_i(a)$ to store optimum cost from x_1 to x_i
3. Preserves equivalence by "cost shifting" [Sch00; Sch76]

Enhanced propagation

Communication between functions goes through functions.

Equivalence Preserving Transformation

Arc EPT

- A cost function c_S, here c_{ij}.
- EPT `Project` $(\{ij\}, \{i\}, a, \alpha)$ shifts cost α between $c_i(a)$ and the cost function c_{ij}.
- projection $(\alpha \geq 0)$, extension $(\alpha < 0)$.

Precondition: $-c_i(a) \leq \alpha \leq \min_{t' \in D^{ij}, t'[i]=a} c_{ij}(t')$;

Procedure `Project`$(\{i,j\}, \{i\}, a, \alpha)$

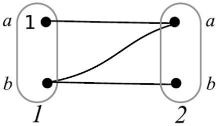
$c_i(a) \leftarrow c_i(a) \oplus \alpha$;

foreach *($t' \in D^{ij}$ such that $t'[i] = a$)* **do**

  $c_{ij}(t') \leftarrow c_{ij}(t') \ominus \alpha$;

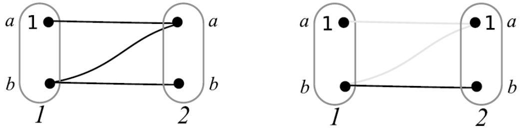
end

Example



$\texttt{Project}(\{1,2\},\{2\},a,1)$

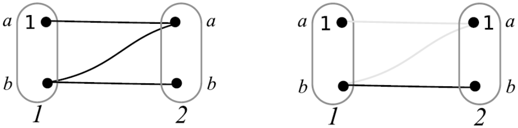
$\rightarrow$



Example

$\texttt{Project}(\{1,2\},\{2\},a,1)$

$\rightarrow$

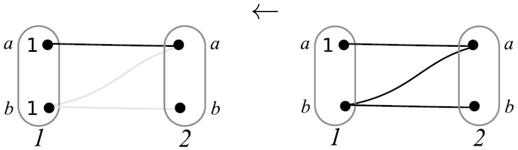


$\leftarrow$

$\texttt{Project}(\{1,2\},\{2\},a,-1)$

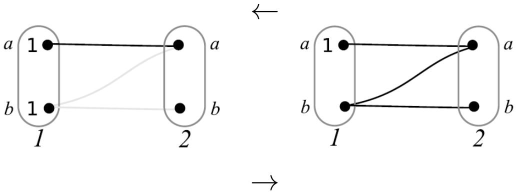
Example

$\texttt{Project}(\{1,2\},\{1\},b,1)$



Example

Project({1, 2}, {1}, b, 1)

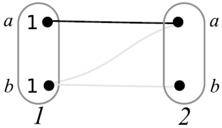


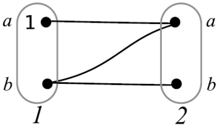
Project({1, 2}, {1}, b, −1)

Example

$\texttt{Project}(\{1,2\},\{1\},b,1)$

$\leftarrow$



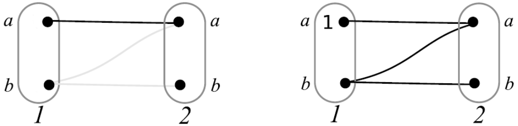


$\Downarrow \qquad \texttt{Project}\ (\{1\},\varnothing,[],1)$

Example

$\texttt{Project}(\{1,2\},\{1\},b,1)$

$\leftarrow$



$\Downarrow \quad \texttt{Project}\ (\{1\},\varnothing,[],1)$

$c_{\varnothing} = 1$

Properties

- Solves tree structured problems (proper ordering), optimum available in $c_{\varnothing}$
- is a reformulation so incremental
- has internal incrementality (supports)
- May loop indefinitely on cyclic graphs
- No unique fixpoint when it exists

Breaking the loops

1. Arc consistency $O(ed^3)$: prevent loops at the arc level [Sch00]
2. Node consistency [Lar02]
3. Directional AC $O(ed^2)$: prevent loops at a global level [Coo03; LS03; LS04]
4. Combine AC and DAC into FDAC [LS03; LS04]
5. Pool costs from all stars to $c_\varnothing$ in EAC [Lar+05]
6. Combine AC+DAC+EAC in EDAC [Lar+05]

All $O(ed)$ space.

Comparison with AC

1. AC, FDAC and EDAC equivalent to classical AC (constraints).
2. DAC equivalent to classical DAC (constraints).
3. AC $<$ FDAC $<$ EDAC in terms of *lb* strength.

Comparison with AC

1. AC, FDAC and EDAC equivalent to classical AC (constraints).
2. DAC equivalent to classical DAC (constraints).
3. AC $<$ FDAC $<$ EDAC in terms of *lb* strength.

Can be enforced on global cost functions too (by emulating DP, or using graph algorithms) [LL10; LL12; Boi+12].

Finding an optimal order [CS04]

Finding an optimal sequence of integer arc EPTs that maximizes the lower bound is NP-hard.

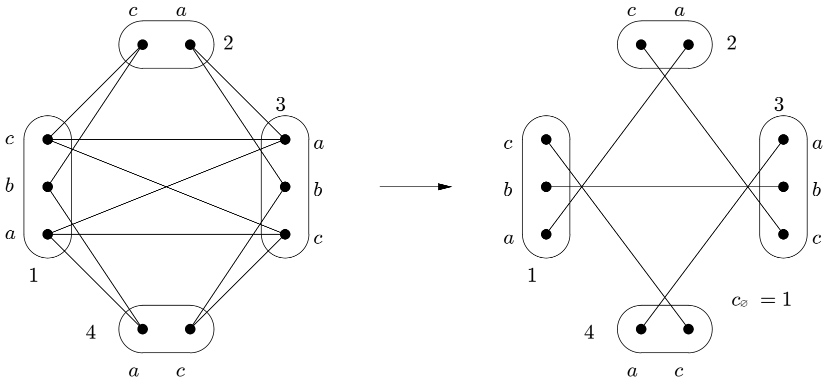
Finding an optimal order [CS04]

Finding an optimal sequence of integer arc EPTs that maximizes the lower bound is NP-hard.

Finding an optimal set[CGS07]

Finding an optimal *set* of *rational* arc EPTs that maximizes the lower bound is in P.
This is achieved by solving an LP (OSAC, finite costs, $k = \infty$).

Reformulation by OSAC



01 LP Variables, for a binary CFN

1. u_i: amount of cost shifted from c_i to $c_\varnothing$
2. p_{ija}: amount of cost shifted from c_{ij} to $a \in D^i$
3. p_{jib}: amount of cost shifted from c_{ij} to $b \in D^j$

See [Sch76; Kos99; CGS07; Wer07; Coo+10].

Optimal Soft Arc Consistency (finite costs, $k = \infty$)

01 LP Variables, for a binary CFN

1. u_i: amount of cost shifted from c_i to $c_\varnothing$
2. p_{ija}: amount of cost shifted from c_{ij} to $a \in D^i$
3. p_{jib}: amount of cost shifted from c_{ij} to $b \in D^j$

OSAC

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

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

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

See [Sch76; Kos99; CGS07; Wer07; Coo+10].

The MRF local polytope [Wer07]

$$\text{Minimize} \sum_{i,a} c_i(a) \cdot x_{ia} + \sum_{\substack{c_{ij} \in C \\ a \in D^i, b \in D^j}} c_{ij}(a,b) \cdot y_{iajb} \quad \text{s.t}$$

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

$$\sum_{b \in D^j} y_{iajb} - x_{ia} = 0 \qquad \forall c_{ij} \in C, \forall a \in D^i \tag{2}$$

$$\sum_{a \in D^i} y_{iajb} - x_{jb} = 0 \qquad \forall c_{ij} \in C, \forall b \in D^j \tag{3}$$

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

u_i multiplier for (1) and p_{ija}/p_{jib} for (2) and (3) (as $\geq$ inequalities).

Duality

We are looking for multipliers u_i and p_{ija} that

1. define a linear inequality with multiplicative constants lower than in the primal criteria (dual constraints)
2. such that the rhs of the inequality (lower bound) is maximum

Duality

We are looking for multipliers u_i and p_{ija} that

1. define a linear inequality with multiplicative constants lower than in the primal criteria (dual constraints)
2. such that the rhs of the inequality (lower bound) is maximum

Dual

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

$$u_i - \sum_{(c_{ij} \in C)} p_{ija} \leq c_i(a) \qquad \forall i \in \{1, \ldots, n\}, \ \forall a \in D^i$$

$$p_{ija} + p_{jib} \leq c_{ij}(a, b) \qquad \forall c_{ij} \in C, \forall (a, b) \in D^{ij}$$

- The local polytope and its dual have been intensely studied, starting with the "Ukrainian" school [Sch76; KK75; KS76; Wer07].

- The local polytope and its dual have been intensely studied, starting with the "Ukrainian" school [Sch76; KK75; KS76; Wer07].
- A variety of non-smooth convex optimization algorithms have been tried with the hope of "faster than LP" resolution [SG07; KPT07; Sav+11; KSS12].

Graphical model polytopes

- The local polytope and its dual have been intensely studied, starting with the "Ukrainian" school [Sch76; KK75; KS76; Wer07].
- A variety of non-smooth convex optimization algorithms have been tried with the hope of "faster than LP" resolution [SG07; KPT07; Sav+11; KSS12].
- [PW15] showed that any "normal" LP can be reduced to such a polytope in linear time (constructive proof).

Have we been doing LP w/o knowing ?

1. Somewhat: AC, DAC, FDAC, EDAC can be seen as approximate greedy Block Coordinate Descent solvers of this dual LP.
2. They all find feasible (but usually non optimal) solutions of the dual.
3. But an optimal bound is not necessarily ideal (OSAC).
4. AC variants all directly deal with finite k or infinite costs.

Can we organize our EPTs better w/o LP?

Bool(P) [Coo+08]

Given a CFN $P = (X, D, C, k)$, Bool(P) is the CSP $(X, D, C - \{c_\varnothing\}, 1)$.

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

Bool(P) [Coo+08]

Given a CFN $P = (X, D, C, k)$, Bool(P) is the CSP $(X, D, C - \{c_\varnothing\}, 1)$.

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

Virtual AC

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

Can we organize our EPTs better w/o LP?

Bool(P) [Coo+08]

Given a CFN $P = (X, D, C, k)$, Bool(P) is the CSP $(X, D, C - \{c_\varnothing\}, 1)$.

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

Virtual AC

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

Virtual AC

Same fixpoint as a variety of converging reformulating BP algorithms in MRF: TRW-S [Kol06], MPLP1[Son+12], SRMP [Kol15], Max-Sum diffusion [KK75; Coo+10], Aug-DAG[KS76]. . .

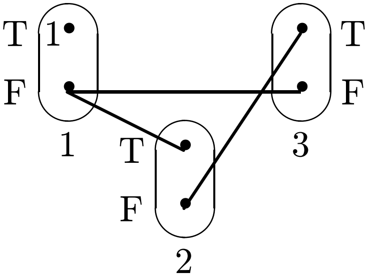
How do we enforce VAC ?

How do we enforce VAC ?

OSAC does it, but without LP

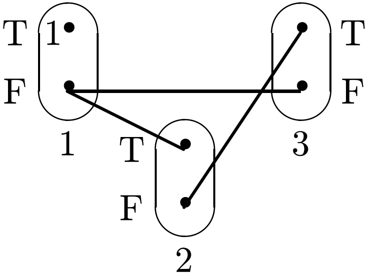
1. Enforce AC in Bool(P) until a wipe-out occurs (record EPTs)
2. Extract a minimal set of EPTs sufficient for the wipe-out
3. Apply cost EPTs on P using suitable cost moves

A “simple” example

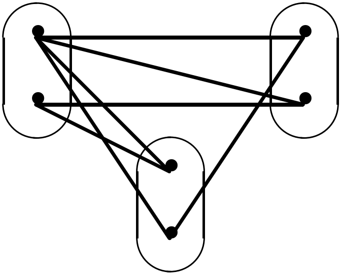


Original problem

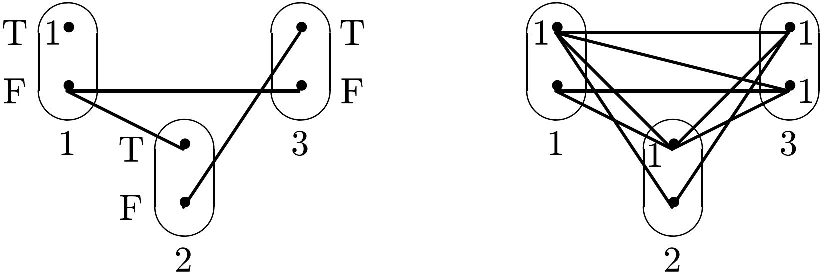
A “simple” example



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

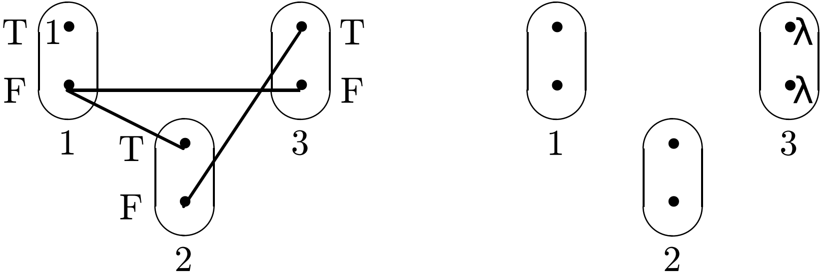


A "simple" example



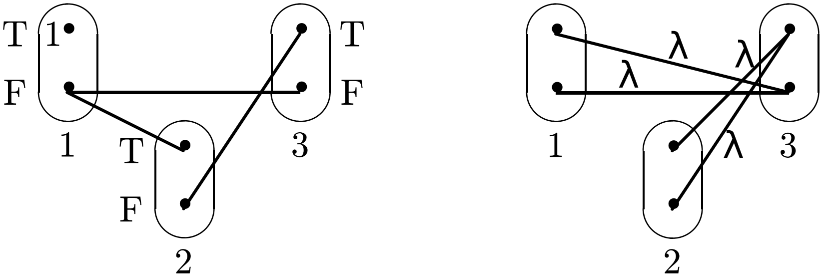
AC: deleting $(3, T)$: wipe out with 3 EPTs !

A “simple” example



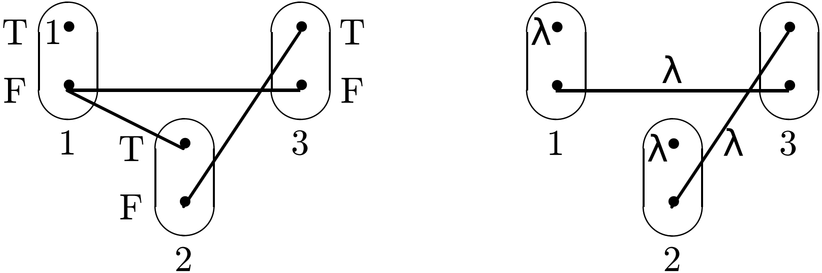
We want to bring λ cost unit to x_3, λ unknown.

A “simple” example

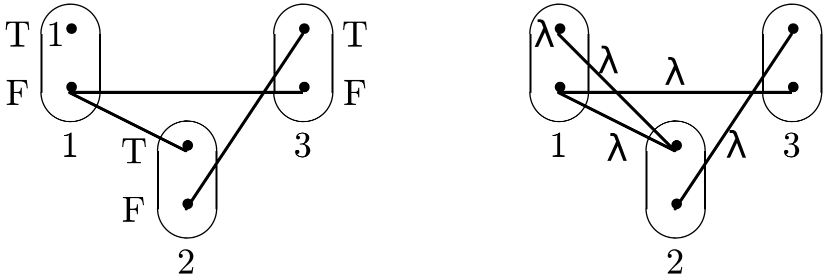


This requires λ virtual cost that needs to be paid by concrete costs. . .

A "simple" example

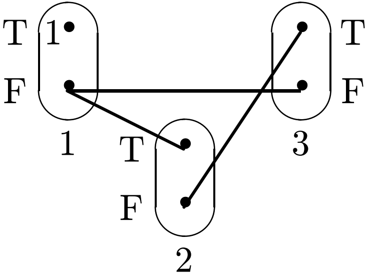


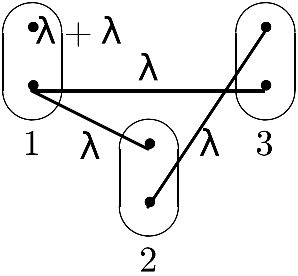
This requires λ virtual cost that needs to be paid by concrete costs... or propagated through EPTs



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

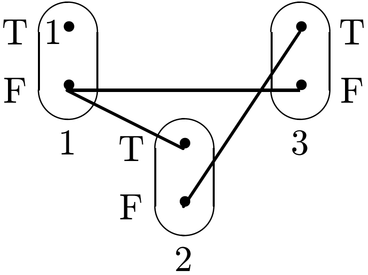
A "simple" example

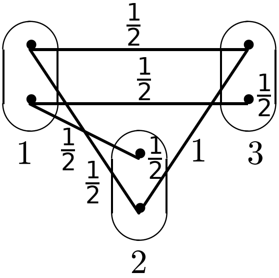




we need 2λ on $(1, T)$ and have only 1 unit of cost: $\lambda = \frac{1}{2}$

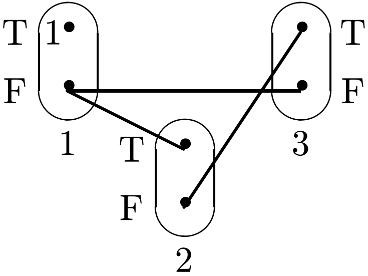
A "simple" example

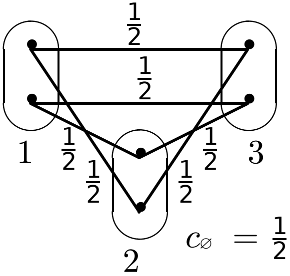




We replay the EPTs using the values of λ

A "simple" example





At the end we are able to project λ to $c_\varnothing$

Complexity

Table cost functions

1. Each iteration is in $O(ed^r)$.
2. May require an infinite number of iterations.
3. ε-convergence in $O(ed^r.k/\varepsilon)$
4. can be much faster than OSAC
5. often accelerates CPLEX on local polytopes

Connection with AC in CSP

Virtual AC

1. solves tree-structured problems,
2. solves CFNs with submodular cost functions (Monge)
3. solves CFNs for which AC is a decision procedure in Bool(P).

Connection with AC in CSP

Virtual AC

1. solves tree-structured problems,
2. solves CFNs with submodular cost functions (Monge)
3. solves CFNs for which AC is a decision procedure in Bool(P).

1. Any solution of Bool(P) has cost $c_\varnothing$ and is therefore optimal.
2. A problem which is VAC and has only one value a in each domain such that $c_i(a) = 0$ is solved.
3. There is always at least one such value (or else not VAC).

Any connection with a famous graph algorithm ?

Boolean binary CFN - QPBO - WMax2SAT

1. Bool(P) is 2-SAT (in P).
2. Minimal propagation DAG made of disjoint paths.
3. Related to Ford-Fulkerson (specific graph),
4. Similar to the "roof-dual" lower bound of QPBO (LP or flow based [BH02])
5. Similar to 'Graph Cut" for binary pairwise supermodular MRF (flow based [KR07])
6. naturally incremental, thanks to EPTs.

Any connection with a famous graph algorithm ?

Boolean binary CFN - QPBO - WMax2SAT

1. Bool(P) is 2-SAT (in P).
2. Minimal propagation DAG made of disjoint paths.
3. Related to Ford-Fulkerson (specific graph),
4. Similar to the "roof-dual" lower bound of QPBO (LP or flow based [BH02])
5. Similar to 'Graph Cut" for binary pairwise supermodular MRF (flow based [KR07])
6. naturally incremental, thanks to EPTs.

Anything similar to VAC in OR/Graph algorithms ?

Implementations

1. black box solver (à la SAT/01LP)

1. black box solver (à la SAT/01LP)
2. table cost functions (tables, lists)

1. black box solver (à la SAT/01LP)
2. table cost functions (tables, lists)
3. global cost functions (Weighted AllDiff, GCC, Regular...)

1. black box solver (à la SAT/01LP)
2. table cost functions (tables, lists)
3. global cost functions (Weighted AllDiff, GCC, Regular...)
4. (treewidth aware) DFBB and Hybrid BFS [All+15]

1. black box solver (à la SAT/01LP)
2. table cost functions (tables, lists)
3. global cost functions (Weighted AllDiff, GCC, Regular. . .)
4. (treewidth aware) DFBB and Hybrid BFS [All+15]
5. Updated Optimality gap (HBFS), anytime behavior

1. black box solver (à la SAT/01LP)
2. table cost functions (tables, lists)
3. global cost functions (Weighted AllDiff, GCC, Regular...)
4. (treewidth aware) DFBB and Hybrid BFS [All+15]
5. Updated Optimality gap (HBFS), anytime behavior
6. Default clever horizontal (value) ordering

1. black box solver (à la SAT/01LP)
2. table cost functions (tables, lists)
3. global cost functions (Weighted AllDiff, GCC, Regular...)
4. (treewidth aware) DFBB and Hybrid BFS [All+15]
5. Updated Optimality gap (HBFS), anytime behavior
6. Default clever horizontal (value) ordering
7. Weighted degree + last conflict vertical ordering heuristics

mulcyber.toulouse.inra.fr/projects/toulbar2

1. black box solver (à la SAT/01LP)
2. table cost functions (tables, lists)
3. global cost functions (Weighted AllDiff, GCC, Regular. . .)
4. (treewidth aware) DFBB and Hybrid BFS [All+15]
5. Updated Optimality gap (HBFS), anytime behavior
6. Default clever horizontal (value) ordering
7. Weighted degree + last conflict vertical ordering heuristics
8. Maintains NC, AC, DAC, FDAC, EDAC and VAC.

1. black box solver (à la SAT/01LP)
2. table cost functions (tables, lists)
3. global cost functions (Weighted AllDiff, GCC, Regular...)
4. (treewidth aware) DFBB and Hybrid BFS [All+15]
5. Updated Optimality gap (HBFS), anytime behavior
6. Default clever horizontal (value) ordering
7. Weighted degree + last conflict vertical ordering heuristics
8. Maintains NC, AC, DAC, FDAC, EDAC and VAC.
9. Maintains non-dominance (aka substitutability aka DEE)

1. black box solver (à la SAT/01LP)
2. table cost functions (tables, lists)
3. global cost functions (Weighted AllDiff, GCC, Regular...)
4. (treewidth aware) DFBB and Hybrid BFS [All+15]
5. Updated Optimality gap (HBFS), anytime behavior
6. Default clever horizontal (value) ordering
7. Weighted degree + last conflict vertical ordering heuristics
8. Maintains NC, AC, DAC, FDAC, EDAC and VAC.
9. Maintains non-dominance (aka substitutability aka DEE)
10. (On the fly) Variable elimination (degree ≤ 3)

1. black box solver (à la SAT/01LP)
2. table cost functions (tables, lists)
3. global cost functions (Weighted AllDiff, GCC, Regular...)
4. (treewidth aware) DFBB and Hybrid BFS [All+15]
5. Updated Optimality gap (HBFS), anytime behavior
6. Default clever horizontal (value) ordering
7. Weighted degree + last conflict vertical ordering heuristics
8. Maintains NC, AC, DAC, FDAC, EDAC and VAC.
9. Maintains non-dominance (aka substitutability aka DEE)
10. (On the fly) Variable elimination (degree ≤ 3)
11. Local search upper bounding (INCOP [NT03])

1. black box solver (à la SAT/01LP)
2. table cost functions (tables, lists)
3. global cost functions (Weighted AllDiff, GCC, Regular...)
4. (treewidth aware) DFBB and Hybrid BFS [All+15]
5. Updated Optimality gap (HBFS), anytime behavior
6. Default clever horizontal (value) ordering
7. Weighted degree + last conflict vertical ordering heuristics
8. Maintains NC, AC, DAC, FDAC, EDAC and VAC.
9. Maintains non-dominance (aka substitutability aka DEE)
10. (On the fly) Variable elimination (degree ≤ 3)
11. Local search upper bounding (INCOP [NT03])
12. Table cost function decomposition

1. black box solver (à la SAT/01LP)
2. table cost functions (tables, lists)
3. global cost functions (Weighted AllDiff, GCC, Regular. . .)
4. (treewidth aware) DFBB and Hybrid BFS [All+15]
5. Updated Optimality gap (HBFS), anytime behavior
6. Default clever horizontal (value) ordering
7. Weighted degree + last conflict vertical ordering heuristics
8. Maintains NC, AC, DAC, FDAC, EDAC and VAC.
9. Maintains non-dominance (aka substitutability aka DEE)
10. (On the fly) Variable elimination (degree ≤ 3)
11. Local search upper bounding (INCOP [NT03])
12. Table cost function decomposition
13. Parallel VNS search [Oua+14]

1. First/second in approximate graphical model MRF/MAP challenges (2010, 2012, 2014).
2. Bioinformatics: pedigree debugging [SGS08], Haplotyping (QTLMap), structured RNA gene finding [ZGS08], Computational Protein Design [Tra+13] (now in OSPREY)
3. RLFAP: closed all CELAR min-interference RLFAP instances fap.zib.de/problems/CALMA
4. Inductive Logic Programming [AR07], Natural Langage Processing (in hltdi-l3), Multi-agent and cost-based planning [KZ10; CRR11], Model Abstraction [SFN11], diagnostic [MJS11b], Music processing and Markov Logic [PT12; PT13], Data mining [MLC13], Partially observable Markov Decision Processes [Dib+13], Probabilistic counting [Erm+13] and inference [MJS11a], . . .

Other solvers

Mostly MRF targeted

- daoopt (exact, DFBB + Treewidth + minibuckets)[a]
- MPLP[b], SRMP[c] : primal/dual like solvers using BCD-based approximate LP bounds for dual and heuristic from primal.
- OpenGM-2[d]: an impressive MRF processing library with many MRF processing algorithms (includes daoopt and many other published algorithms, exact or not. Toulbar2 soon).

---

[a] github.com/lotten/daoopt
[b] cs.nyu.edu/ dsontag/code/README_v2.html
[c] pub.ist.ac.at/ vnk/software.html
[d] hci.iwr.uni-heidelberg.de/opengm2

Application to Computational Protein Design

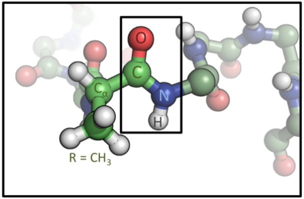
Joint work with D. Allouche, Isabelle André (LISBP-INSA), Sophie Barbe (LISBP-INSA), Jessica Davies, Simon de Givry, George Katsirelos, Barry O'Sullivan (Insight Centre, Ireland), Steve Prestwich (Insight Centre), David Simoncini, Seydou Traoré (LISBP-INSA).

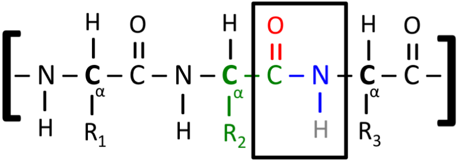
What is a protein ?

(Kudos to wikipedia)

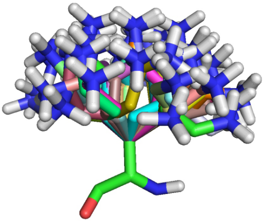
Amino acids, proteins

- Proteins are linear chains of amino-acids (20 natural AAs).
- All AAs share a common "core" and have a variable side-chain.





Side-chains are flexible (ARG)



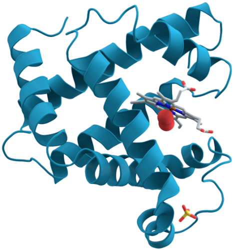
Why ?

- Proteins have various functions in the cell: catalysis, signaling, recognition, regulation. . .
- Efficient, biodegrable, 10^6 to 10^{20} speedups
- Some reactions / ligands miss enzymes / partners.
- Medecine, cosmetics, food, bio-energies. . .
- Nano-technologies (shape more than function).

Protein function linked to its 3D shape through its amino acid composition.

Protein design's aim

Identify sequences that have a suitable function (shape).



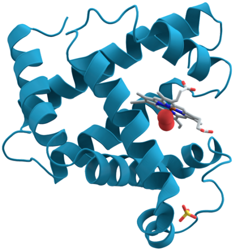
Protein function linked to its 3D shape through its amino acid composition.

Protein design's aim

Identify sequences that have a suitable function (shape).

Issue

There are 20^n proteins of length n.
Impossible to synthesize and test all of them.



The CPD problem - stability variant

Preparation

- A backbone is chosen/built from a known protein/structure (or *de novo*).
- Positions are set as mutable, flexible or rigid
- The aim is to find an AA sequence that folds, stably, in the backbone.

The CPD problem - stability variant

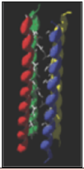
Preparation

- A backbone is chosen/built from a known protein/structure (or *de novo*).
- Positions are set as mutable, flexible or rigid
- The aim is to find an AA sequence that folds, stably, in the backbone.

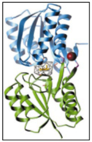
Issues

- CPD is a sort of inverse of folding.
- But folding is far from being a solved problem

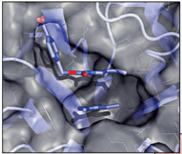
Successes of Protein Design



- Coiled-coils *Harbury et al (Science 1998)*

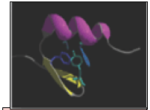


- Sensors & receptors toward non native ligands *Hellinga et al (Nature, 2003)*



2010

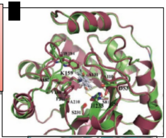
Design of Biocatalyst for a stereoselective & bimolecular reaction (Diels-alder)
Siegel et al (Science 2010)



- Zinc-finger *Dahiyat et al (Science 1997)*

2008

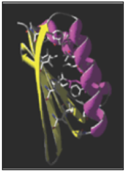
- New catalyst for a multi-step-reaction (retro-aldolase) *Jiang et al (Science 2008)*



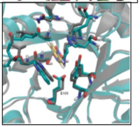
2003

1997

- A novel topology (Top7) *Khulman et al (Science 2003)*



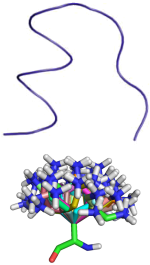
- De novo design of new functional enzyme (Kemp elimination) *Rothlisberger et al (Nature 2008)*



The (basic) CPD problem: search space

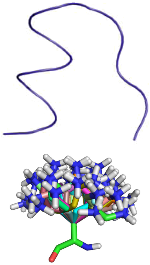
Rigid backbone variant

1. Assume a rigid protein backbone.
2. Choose 1 AA among possible ones at each mutable position.
3. Spatial conformation discretized in rotamers.
4. Statistically frequent orientations.
5. Several 100's rotamers per position.



Rigid backbone variant

1. Assume a rigid protein backbone.
2. Choose 1 AA among possible ones at each mutable position.
3. Spatial conformation discretized in rotamers.
4. Statistically frequent orientations.
5. Several 100's rotamers per position.



Search Space

1. Fully discrete description, defined by a choice of rotamer (AA $\times$ conformation) for each position.
2. Search space can be $\approx 250^n$

Stable = minimum energy (GMEC, NP-hard [PW02])

Energy: interactions between atoms.

- Electrostatic, van der Waals (Amber)
- Dihedral torsion angles, Implicit Solvation (EEF1)
- "Statistical terms" (Talaris)
- Cutoff functions

Stable = minimum energy (GMEC, NP-hard [PW02])

Energy: interactions between atoms.

- Electrostatic, van der Waals (Amber)
- Dihedral torsion angles, Implicit Solvation (EEF1)
- "Statistical terms" (Talaris)
- Cutoff functions

Pairwise decomposable energy

- backbone/backbone (constant)
- backbone/rotamer (depends on rotamer)
- rotamer/rotamer (depends on pairs of rotamers)

Stable = minimum energy (GMEC, NP-hard [PW02])

Energy: interactions between atoms.

- Electrostatic, van der Waals (Amber)
- Dihedral torsion angles, Implicit Solvation (EEF1)
- "Statistical terms" (Talaris)
- Cutoff functions

Pairwise decomposable energy

- backbone/backbone (constant)
- backbone/rotamer (depends on rotamer)
- rotamer/rotamer (depends on pairs of rotamers)

$$E(c) = E_{\varnothing} + \sum_{i=1}^{n} E(i_r) + \sum_{i<j} E(i_r, j_s)$$

Dedicated CPD Methods

Dominance / Sustitutability / Dead End Elimination [Des+92]

$$E(i_a) + \sum_{j \neq i}^{n} \min_{c} E(i_a, j_c) > E(i_b) + \sum_{j \neq i}^{n} E \max_{b} E(i_b, j_c)$$

Dedicated CPD Methods

Dominance / Sustitutability / Dead End Elimination [Des+92]

$$E(i_a) + \sum_{j \neq i}^{n} \min_c E(i_a, j_c) > E(i_b) + \sum_{j \neq i}^{n} E \max_b E(i_b, j_c)$$

Strengthened by [Gol94]

$$E(i_a) - E(i_b) + \sum_{j \neq i}^{n} \min_c \left[E(i_a, j_c) - E(i_b, j_c)\right] > 0$$

Dedicated CPD Methods

Dominance / Sustitutability / Dead End Elimination [Des+92]

$$E(i_a) + \sum_{j \neq i}^{n} \min_{c} E(i_a, j_c) > E(i_b) + \sum_{j \neq i}^{n} E \max_{b} E(i_b, j_c)$$

Strenghtened by [Gol94]

$$E(i_a) - E(i_b) + \sum_{j \neq i}^{n} \min_{c} \left[E(i_a, j_c) - E(i_b, j_c) \right] > 0$$

Many further enhancements (splitting, pairs...). Polynomial time pre-processing.

Dedicated CPD Methods

Dominance / Sustitutability / Dead End Elimination [Des+92]

$$E(i_a) + \sum_{j \neq i}^{n} \min_{c} E(i_a, j_c) > E(i_b) + \sum_{j \neq i}^{n} E \max_{b} E(i_b, j_c)$$

Strenghtened by [Gol94]

$$E(i_a) - E(i_b) + \sum_{j \neq i}^{n} \min_{c} \left[E(i_a, j_c) - E(i_b, j_c) \right] > 0$$

Many further enhancements (splitting, pairs...). Polynomial time pre-processing.

"(Soft) substitutability" [Coo97; LRD12]
Dominating 1-clause rule in MaxSAT [NR00].

polytime DEE, GMEC NP-hard

- DEE cannot reduce all domains to singletons
- Followed by A^* best-first search using the following lower bound (admissible heuristics) [GLD08]:

$$\underbrace{\sum_{i=1}^{d} E(i_r) + \sum_{j=i+1}^{d} E(i_r, j_s)}_{\text{Assigned}} + \textstyle\sum_{j=d+1}^{n} \Big[\underbrace{\min_s (E(j_s) + \sum_{i=1}^{d} E(i_r, j_s)}_{\text{Forward checking}} + \underbrace{\sum_{k=j+1}^{n} \min_u E(j_s, k_u))}_{\text{DAC counts}} \Big]$$

polytime DEE, GMEC NP-hard

- DEE cannot reduce all domains to singletons
- Followed by A^* best-first search using the following lower bound (admissible heuristics) [GLD08]:

$$\underbrace{\sum_{i=1}^{d} E(i_r) + \sum_{j=i+1}^{d} E(i_r, j_s)}_{\text{Assigned}} + \sum_{j=d+1}^{n} \Big[\underbrace{\min_s (E(j_s) + \sum_{i=1}^{d} E(i_r, j_s)}_{\text{Forward checking}} + \underbrace{\sum_{k=j+1}^{n} \min_u E(j_s, k_u))}_{\text{DAC counts}} \Big]$$

Lower bound

- Same as a lower bound introduced in AI (WCSP) in 1994 [Wal95].
- Obsoleted by local consistencies.

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

Our targets [All+14]

- Identify a most efficient model/solving technique for the rigid backbone/rotamer based/pairwise energy CPD problem.
- Do one of the first large spectrum comparison of NP-complete optimization techniques (AI: CFN, CP, SAT, MRF and OR: ILP, QP, QPBO) on one well defined, important optimization problem.
- Learn from it.

Partial Weighted maxSAT

PW MaxSAT

- Boolean variables, litteral: variable or its negation
- Weighted clauses: disjunction ($\vee$) of litterals.
- criteria: sum of weight of violated clauses.
- B&B - Core solvers: MiniMaxSat [HLO08],akMaxSat [Kue10] - bincd [HMM11],wpm1/2 [ABL09; ABL10],MaxHS [DB13]

Partial Weighted maxSAT

PW MaxSAT

- Boolean variables, litteral: variable or its negation
- Weighted clauses: disjunction ($\vee$) of litterals.
- criteria: sum of weight of violated clauses.
- B&B - Core solvers: MiniMaxSat [HLO08],akMaxSat [Kue10] - bincd [HMM11],wpm1/2 [ABL09; ABL10],MaxHS [DB13]

Direct encoding

- d_{i_a}: use i_a
- $\forall i_r, i_s, i_r \neq i_s, (\neg d_{i_r} \vee \neg d_{i_s})$ (AMO)
- $\forall i, (\bigvee_r d_{i_r})$ (ALO)
- $(\neg d_{i_r}, E(i_r)$ and $(\neg d_{i_r} \vee \neg d_{j_s}, E(i_r, j_s))$

Tuple encoding

Property [Bac07]

In CSP, Unit Propagation on this encoding enforces AC on the CSP. Close to the local polytope ILP model.

Direct encoding

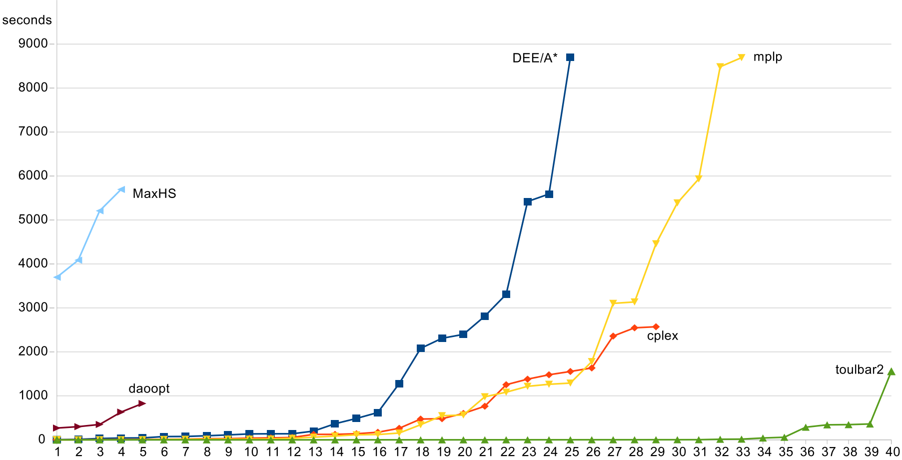
- d_{i_a} + AMO + ALO.
- $p_{i_r j_s}$: pair i_a, j_s is used.
- $\forall i_r, j_s : (d_{i_r} \vee \neg p_{i_r j_s})$ and $(d_{j_s} \vee \neg p_{i_r j_s})$.
- $\forall i_r, j(\neg d_{i_r} \vee \bigvee_s p_{i_r j_s})$
- idem for $E(i_r)$, $\forall i_r, j_s(\neg p_{i_r j_s}, E(i_r, j_s))$

A realistic benchmark: 35+12 designs tested

The designs

1. Extracted from the litterature [Tra+13],
2. Good resolution of the PDB structures,
3. Structure preparation,
4. Domains assigned based on accessibility,
5. Amber + EEF1 + No cutoff (almost complete graphs)
6. Variable search space size, from 10^{26} to 10^{249}

Results - 9000 seconds



Analysis

1. **QP by Cplex**: dense model, but weak and somewhat expensive lb (very large node file, large gaps).

From failures. . .

Analysis

1. **QP by Cplex**: dense model, but weak and somewhat expensive lb (very large node file, large gaps).
2. **SDP based QPO**: probably tight lower bound, but far too expensive (few nodes explored after several hours). biqmac library of MaxCut beasley instances size 100: solved in 1" by tb2, 1' by biqmac.

From failures. . .

Analysis

1. **QP by Cplex**: dense model, but weak and somewhat expensive lb (very large node file, large gaps).
2. **SDP based QPO**: probably tight lower bound, but far too expensive (few nodes explored after several hours). biqmac library of MaxCut beasley instances size 100: solved in 1" by tb2, 1' by biqmac.
3. **MaxSAT, direct**: branch and bound solvers very fast (36k nodes/sec, 100 times faster than tb2). found incumbent solutions but never started the optimality proof. Weak lb (root = 25% of optimum, tb2 always > 97%).

From failures...

Analysis

1. **QP by Cplex**: dense model, but weak and somewhat expensive lb (very large node file, large gaps).
2. **SDP based QPO**: probably tight lower bound, but far too expensive (few nodes explored after several hours). biqmac library of MaxCut beasley instances size 100: solved in 1" by tb2, 1' by biqmac.
3. **MaxSAT, direct**: branch and bound solvers very fast (36k nodes/sec, 100 times faster than tb2). found incumbent solutions but never started the optimality proof. Weak lb (root = 25% of optimum, tb2 always $>$ 97%).
4. **MaxSAT, tuple**: b&b,strong lower bound (should be similar to VAC for core based solvers). Still weaker than tb2 and very slow (2 nodes before timeout at best for akmaxsat). No incumbent. Core based better (maxHS, good lb).

Analysis

1. **Daoopt**: almost complete graphs. Not ideal for tree decomposition based methods.

Analysis

1. **Daoopt**: almost complete graphs. Not ideal for tree decomposition based methods.
2. **DEE/A***: surprisingly good given the lower bound used. Very strong preprocessing.

Analysis

1. **Daoopt**: almost complete graphs. Not ideal for tree decomposition based methods.
2. **DEE/A***: surprisingly good given the lower bound used. Very strong preprocessing.
3. **ILP - Cplex**: LP bound similar to OSAC (dual). tb2 has upper bounding (k). Similar number of nodes but tb2 much faster (ILP: 1 to 40 nodes / minutes, tb2: 1 to 40 thousand).

Analysis

1. **Daoopt**: almost complete graphs. Not ideal for tree decomposition based methods.
2. **DEE/A***: surprisingly good given the lower bound used. Very strong preprocessing.
3. **ILP** - **Cplex**: LP bound similar to OSAC (dual). tb2 has upper bounding (k). Similar number of nodes but tb2 much faster (ILP: 1 to 40 nodes / minutes, tb2: 1 to 40 thousand).
4. **MPLP**: no branching but able to solve few more problems than CPLEX.

Analysis

1. **Daoopt**: almost complete graphs. Not ideal for tree decomposition based methods.
2. **DEE/A***: surprisingly good given the lower bound used. Very strong preprocessing.
3. **ILP** - **Cplex**: LP bound similar to OSAC (dual). tb2 has upper bounding (k). Similar number of nodes but tb2 much faster (ILP: 1 to 40 nodes / minutes, tb2: 1 to 40 thousand).
4. **MPLP**: no branching but able to solve few more problems than CPLEX.

A Lesson for (AI) Optimization

The lower bounding/search efforts compromise is, AFAIK, not understood, nor exploited. But may be crucial.

Final note and Acknowledgments

This is all for a rigid backbone. Modern CPD increasingly uses "flexible" representations (eg. with a backbone ensemble).

Final note and Acknowledgments

This is all for a rigid backbone. Modern CPD increasingly uses "flexible" representations (eg. with a backbone ensemble).

Thanks to...

- Tomas Werner (Center for Machine Perception, Praha, Czech rep.) for contributing to the WCSP/CFN/MRF/LP connection.
- Bruce Donald and Kyle Roberts (Duke Univ., USA) for the open source software Osprey and helping us with it.

Final note and Acknowledgments

This is all for a rigid backbone. Modern CPD increasingly uses "flexible" representations (eg. with a backbone ensemble).

Thanks to...

- Tomas Werner (Center for Machine Perception, Praha, Czech rep.) for contributing to the WCSP/CFN/MRF/LP connection.
- Bruce Donald and Kyle Roberts (Duke Univ., USA) for the open source software Osprey and helping us with it.

Questions ?

References I

Carlos Ansótegui, María Luisa Bonet, and Jordi Levy. "Solving (weighted) partial MaxSAT through satisfiability testing". In: *Theory and Applications of Satisfiability Testing-SAT 2009*. Springer, 2009, pp. 427–440.

Carlos Ansótegui, Maria Luisa Bonet, and Jordi Levy. "A New Algorithm for Weighted Partial MaxSAT." In: *Proceedings of 20th National Conference on Artificial Intelligence (AAAI'10)*. 2010.

David Allouche et al. "Computational protein design as an optimization problem". In: *Artificial Intelligence* 212 (2014), pp. 59–79.

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.

Érick Alphonse and Céline Rouveirol. "Extension of the top-down data-driven strategy to ILP". In: *Inductive Logic Programming*. Springer, 2007, pp. 49–63.

Fahiem Bacchus. "GAC via unit propagation". In: *Principles and Practice of Constraint Programming–CP 2007*. Springer, 2007, pp. 133–147.

References II

Christian Bessiere et al. "Circuit complexity and decompositions of global constraints". In: *arXiv preprint arXiv:0905.3757* (2009).

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

P Boizumault et al. "Filtering Decomposable Global Cost Functions". In: *Proceedings of the National Conference on Artificial Intelligence*. 2012.

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.

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

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

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

References III

M.C. Cooper. "Fundamental properties of neighbourhood substitution in constraint satisfaction problems". In: *Artificial Intelligence* 90.1-2 (1997), pp. 1–24.

Martin C Cooper, Marie de Roquemaurel, and Pierre Régnier. "A weighted CSP approach to cost-optimal planning". In: *Ai Communications* 24.1 (2011), pp. 1–29.

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

Jessica Davies and Fahiem Bacchus. "Exploiting the Power of MIP Solvers in MaxSAT". In: *Theory and Applications of Satisfiability Testing–SAT 2013*. Springer, 2013, pp. 166–181.

J Desmet et al. "The dead-end elimination theorem and its use in protein side-chain positioning." In: *Nature* 356.6369 (Apr. 1992), pp. 539–42. ISSN: 0028-0836. URL: http://www.ncbi.nlm.nih.gov/pubmed/21488406.

Jilles Steeve Dibangoye et al. "Optimally solving Dec-POMDPs as continuous-state MDPs". In: *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*. AAAI Press. 2013, pp. 90–96.

Stefano Ermon et al. "Embed and project: Discrete sampling with universal hashing". In: *Advances in Neural Information Processing Systems*. 2013, pp. 2085–2093.

Eugene C. Freuder. "A sufficient Condition for Backtrack-free Search". In: *Journal of the ACM* 29.1 (1982), pp. 24–32.

Eugene C. Freuder. "A sufficient Condition for Backtrack-Bounded Search". In: *Journal of the ACM* 32.14 (1985), pp. 755–761.

Ivelin Georgiev, Ryan H Lilien, and Bruce R Donald. "The minimized dead-end elimination criterion and its application to protein redesign in a hybrid scoring and search algorithm for computing partition functions over molecular ensembles." In: *Journal of computational chemistry* 29.10 (July 2008), pp. 1527–42. ISSN: 1096-987X. DOI: 10.1002/jcc.20909. URL: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3263346%5C&tool=pmcentrez%5C&rendertype=abstract.

R F Goldstein. "Efficient rotamer elimination applied to protein side-chains and related spin glasses." In: *Biophysical journal* 66.5 (May 1994), pp. 1335–40. ISSN: 0006-3495. DOI: 10.1016/S0006-3495(94)80923-3. URL: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=1275854%5C&tool=pmcentrez%5C&rendertype=abstract.

References V

Federico Heras, Javier Larrosa, and Albert Oliveras. "MiniMaxSAT: An Efficient Weighted Max-SAT solver." In: *J. Artif. Intell. Res.(JAIR)* 31 (2008), pp. 1–32.

Federico Heras, Antonio Morgado, and Joao Marques-Silva. "Core-Guided Binary Search Algorithms for Maximum Satisfiability." In: *Proceedings of 21th National Conference on Artificial Intelligence (AAAI'11)*. 2011.

Frank R Kschischang, Brendan J Frey, and Hans-Andrea Loeliger. "Factor graphs and the sum-product algorithm". In: *Information Theory, IEEE Transactions on* 47.2 (2001), pp. 498–519.

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

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.

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

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.

Nikos Komodakis, Nikos Paragios, and Georgios Tziritas. "MRF optimization via dual decomposition: Message-passing revisited". In: *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*. IEEE. 2007, pp. 1–8.

Vladimir Kolmogorov and Carsten Rother. "Minimizing nonsubmodular functions with graph cuts-a review". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 29.7 (2007), pp. 1274–1279.

VK Koval' and Mykhailo Ivanovich Schlesinger. "Two-dimensional programming in image analysis problems". In: *Avtomatika i Telemekhanika* 8 (1976), pp. 149–168.

Jörg Hendrik Kappes, Bogdan Savchynskyy, and Christoph Schnörr. "A bundle approach to efficient MAP-inference by Lagrangian relaxation". In: *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE. 2012, pp. 1688–1695.

Adrian Kuegel. "Improved exact solver for the weighted Max-SAT problem". In: *Workshop Pragmatics of SAT*. 2010.

References VII

Akshat Kumar and Shlomo Zilberstein. "Point-based backup for decentralized POMDPs: Complexity and new algorithms". In: *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1*. International Foundation for Autonomous Agents and Multiagent Systems. 2010, pp. 1315–1322.

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.

J. Larrosa. "On Arc and Node Consistency in weighted CSP". In: *Proc. AAAI'02*. Edmondton, (CA), 2002, pp. 48–53.

Jimmy Lee and K. L. Leung. "A Stronger Consistency for Soft Global Constraints in Weighted Constraint Satisfaction". In: *AAAI*. Ed. by Maria Fox and David Poole. AAAI Press, 2010.

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.

References VIII

Christophe Lecoutre, Olivier Roussel, and Djamel E Dehani. "WCSP integration of soft neighborhood substitutability". In: *Principles and Practice of Constraint Programming*. Springer. 2012, pp. 406–421.

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.

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

Paul Maier, Dominik Jain, and Martin Sachenbacher. "Compiling AI engineering models for probabilistic inference". In: *KI 2011: Advances in Artificial Intelligence*. Springer, 2011, pp. 191–203.

Paul Maier, Dominik Jain, and Martin Sachenbacher. "Diagnostic hypothesis enumeration vs. probabilistic inference for hierarchical automata models". In: *the International Workshop on Principles of Diagnosis (DX), Murnau, Germany*. 2011.

Jean-Philippe Métivier, Samir Loudni, and Thierry Charnois. "A constraint programming approach for mining sequential patterns in a sequence database". In: *Proceedings of the ECML/PKDD Workshop on Languages for Data Mining and Machine Learning*. arXiv preprint arXiv:1311.6907. Praha, Czech republic, 2013.

Rolf Niedermeier and Peter Rossmanith. "New Upper Bounds for Maximum Satisfiability". In: *J. Algorithms* 36.1 (2000), pp. 63–88.

Bertrand Neveu and Gilles Trombettoni. "Incop: An open library for incomplete combinatorial optimization". In: *Principles and Practice of Constraint Programming–CP 2003*. Springer. 2003, pp. 909–913.

Abdelkader Ouali et al. "Cooperative parallel decomposition guided VNS for solving weighted CSP". In: *Hybrid Metaheuristics*. Springer, 2014, pp. 100–114.

Judea Pearl. *Probabilistic Reasoning in Intelligent Systems, Networks of Plausible Inference*. Palo Alto: Morgan Kaufmann, 1988.

Hélène Papadopoulos and George Tzanetakis. "Modeling Chord and Key Structure with Markov Logic." In: *Proc. Int. Conf. of the Society for Music Information Retrieval (ISMIR)*. 2012, pp. 121–126.

References X

Hélene Papadopoulos and George Tzanetakis. "Exploiting structural relationships in audio music signals using Markov Logic Networks". In: *ICASSP 2013-38th International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Canada (2013)*. 2013, pp. 4493–4497.

Niles A Pierce and Erik Winfree. "Protein design is NP-hard." In: *Protein engineering* 15.10 (Oct. 2002), pp. 779–82. ISSN: 0269-2139. URL: http://www.ncbi.nlm.nih.gov/pubmed/12468711.

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.

Claude-Guy Quimper and Toby Walsh. "Global grammar constraints". In: *Principles and Practice of Constraint Programming-CP 2006*. Springer, 2006, pp. 751–755.

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

J.C. Régin. "Generalized Arc Consistency for Global Cardinality Constraints". In: *Proc. of AAAI'96*. Portland, OR, 1996, pp. 362–367.

Thomas J Richardson and Rüdiger L Urbanke. "The capacity of low-density parity-check codes under message-passing decoding". In: *Information Theory, IEEE Transactions on* 47.2 (2001), pp. 599–618.

Bogdan Savchynskyy et al. "A study of Nesterov's scheme for Lagrangian decomposition and MAP labeling". In: *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE. 2011, pp. 1817–1823.

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.

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.

Peter Struss, Alessandro Fraracci, and D Nyga. "An Automated Model Abstraction Operator Implemented in the Multiple Modeling Environment MOM". In: *25th International Workshop on Qualitative Reasoning, Barcelona, Spain*. 2011.

T. Schiex, H. Fargier, and G. Verfaillie. "Valued Constraint Satisfaction Problems: hard and easy problems". In: *Proc. of the 14th IJCAI*. Montréal, Canada, Aug. 1995, pp. 631–637.

Michail I Schlesinger and VV Giginjak. "Solving (max,+) problems of structural pattern recognition using equivalent transformations". In: *Upravlyayushchie Sistemy i Mashiny (Control Systems and Machines), Kiev, Naukova Dumka* 1 (2007).

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.

David Sontag et al. "Tightening LP relaxations for MAP using message passing". In: *arXiv preprint arXiv:1206.3288* (2012).

Seydou Traoré et al. "A new framework for computational protein design through cost function network optimization". In: *Bioinformatics* 29.17 (2013), pp. 2129–2136.

Willem-Jan Van Hoeve, Gilles Pesant, and Louis-Martin Rousseau. "On global warming: Flow-based soft global constraints". In: *Journal of Heuristics* 12.4-5 (2006), pp. 347–373.

References XIII

R. Wallace. "Directed Arc Consistency Preprocessing". In: *Selected papers from the ECAI-94 Workshop on Constraint Processing*. Ed. by M. Meyer. LNCS 923. Berlin: Springer, 1995, pp. 121–137.

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.

Matthias Zytnicki, Christine Gaspin, and Thomas Schiex. "DARN! A weighted constraint solver for RNA motif localization". In: *Constraints* 13.1-2 (2008), pp. 91–109.