RÉPUBLIQUE
FRANÇAISE
*Liberté
Égalité
Fraternité*

INRAE MIA TOULOUSE

# Toulbar2: optimizing discrete multivariate models

Graphical models &
Constraint programming

Thomas Schiex & Simon de Givry
firstname.name@inrae.fr

github.com/toulbar2/toulbar2 pip3 install pytoulbar2 toulbar2.github.io/toulbar2

ANITI

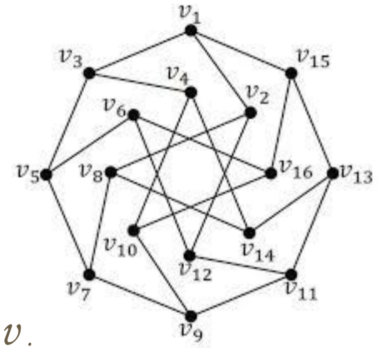
# Modeling: Cost Function Networks (CFN)

- Discrete variables: $X_1,...X_n$
- Joint function on these: $E(X_1,...X_n) = -\log(P(X_1,...,X_n)) + cte$
- $E$ is “infinite-valued” $(E = \infty \asymp false \asymp zero\ probability)$
- Described as the sum of “elementary” functions
  - Cost tensors (space exponential in the number of involved variables)
  - Predefined global functions: *AllDiff*$(X_1,...X_m)$, *Regular*$(A, X_1,.., X_m)$, *Knapsack*$(A,c,X_1,.., X_n)$...

64 bits
fixed point
saturating arithmetics

- Many representable frameworks (many file formats):
  - SAT, weighted MaxSAT, Pseudo-Boolean & 01LP, Q(U)BO, CP/COP (XCSP3)
  - Hidden Markov Models, Markov Random Fields, Bayesian nets (UAI)

# Graphs *(V,E)* & colors *(k)*



- One variable $X_i$ per vertex $i \in V$
- Domains = possible colors
- $k$-coloring : for each $(i,j) \in E$, $f_{ij} \propto$ *eye(k)*
- min-*cost* $k$-coloring : adding $f_i$
- max-$k$-coloring: relaxing $f_{ij}$
- Cost from $f_{ij}$ and $f_i$ are added
- possible separation of costs (Pareto)

$$\begin{array}{c|ccc} & & v_j & \\ \hline & \infty & 0 & 0 \\ v_i & 0 & \infty & 0 \\ & 0 & 0 & \infty \end{array}$$

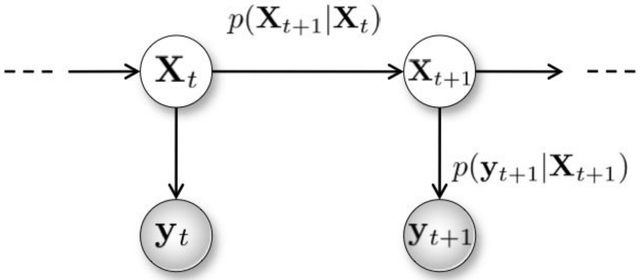
$$\begin{array}{|c|c|c|} \hline 0 & 1 & 2 \\ \hline \end{array}$$

$$\begin{array}{|c|c|c|} \hline 1 & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & 0 & 1 \\ \hline \end{array}$$

```
infinite = 1000000
cfn = pytoulbar2.CFN(infinite)
for i in V: cfn.AddVariable(f'X{i}',range(k))
for (i,j) in E: cfn.AddFunction([f'X{i}',f'X{j}'],infinite*np.eye(k).flatten())
cfn.Solve()
```

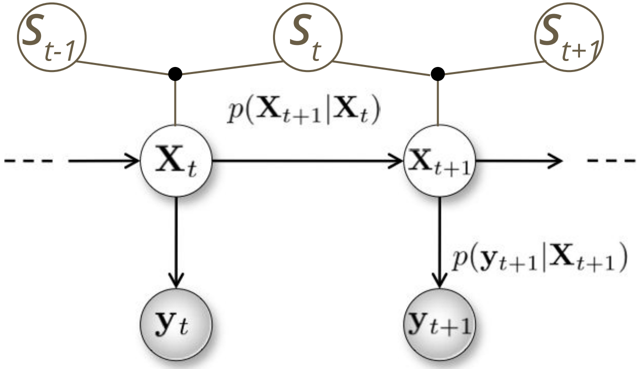
# Hidden Markov Model

- Variables $X_t$ and $Y_t$ with their domains
- Functions $f(X_t, Y_t) = -\log(p(Y_t|X_t))$ and $f(X_{t+1}, X_t) = -\log(p(X_{t+1}|X_t))$
- Treewidth = 1 (acyclic)... dynamic programming
- A specialized Viterbi will be better, but...


$p(\mathbf{X}_{t+1}|\mathbf{X}_t)$
$\mathbf{X}_t$
$\mathbf{X}_{t+1}$
$p(\mathbf{y}_{t+1}|\mathbf{X}_{t+1})$
$\mathbf{y}_t$
$\mathbf{y}_{t+1}$

# Hidden Markov Model

- Variables $X_t$ and $Y_t$ with their domains
- Functions $f(X_t, Y_t) = -\log(p(Y_t|X_t))$ and $f(X_{t+1}, X_t) = -\log(p(X_{t+1}|X_t))$
- Treewidth = 1 (acyclic)... dynamic programming
- The succession of hidden states belongs to a regular language (automata)
- *Regular(A,X₁,...,Xₙ)* decomposable in 3D-tensors $A(S_i, X_i, S_{i+1})$



# Various algorithms for 3 main queries

1. Find $(x_1,\dots x_n)$ minimizing $E(x_1,\dots,x_n)$ decision NP-complete
   a. optimality proof by default (logical reasoning & *reductio ad absurdum*)
   b. anytime, with shrinking optimality gap (predefined or on the fly)
   c. branch & bound with dedicated bounds (generalized CP/SAT inference, cvgt Message Passing)
   d. depth-first or hybrid best/depth first search (default)
   e. can exploit the problem structure (treewidth)
   f. local search solvers (VNS, PILS,...), better solutions faster but...
   g. SDP low rank solver (ICML'22)
   h. C++ (MPI) implementation with Python API (`pytoulbar2`) - Linux/MacOS (Windows soon!)
2. Counting (solutions, partition function) #P-complete
   a. exact algorithms (very expensive except for structured problems)
   b. approximation with deterministic guarantee (same, but tunable)
3. Bi-objective optimization (Pareto front, CPAIOR'24) forgemia.inra.fr/samuel.buchet/tb2_twophase

Winner of Max-CSP/COP (CPAI08, XCSP3 2022, 2023, 2024) and graphical models (UAI 2008, 2010, 2014, 2022 MAP task) challenges.

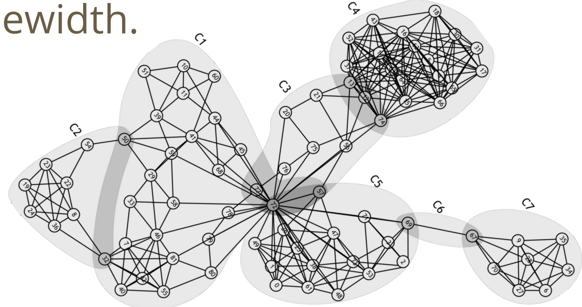
# Max-Cut vs Min-Cut, MRF-based image segmentation

- Graph $(V,E)$ with two colors (vertices partition, symmetry)
- MaxCut: for every $e_{ij} \in E$, $f_{ij} = -\mathbb{1}[X_i \neq X_j]$ (minimization)
  - $f_{ij}$ is supermodular: NP-hard
- MinCut: for every $e_{ij} \in E$, $f_{ij} = \mathbb{1}[X_i \neq X_j]$
  - fij is submodular: polytime (one bound, VAC, gives this polytime behavior)

- Image segmentation: Hidden Markov Random Field, still submodular
  - use dedicated implementations (V. Kolmogorov) for pure segmentation
  - used as the last “layer” of neural architectures for detailed semantic segmentation.

# Radio Link Frequency Assignment (CELAR)

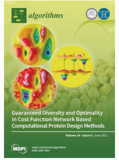
- Generalization of $k$-coloring
- Set of radio links with available frequencies (variables)
- “Nearby” links must use sufficiently different frequencies $f_{ij} = \infty \times \mathbb{1}|X_i\text{-}X_j < k|$
- Extra technological constraints (constant emission/reception frequency shift)
- Criteria:
  - minimize the number of frequencies used (*N-values* global constraint)
  - minimize the number of links subject to interference (replace ∞ by dedicated costs)
- Spatial interactions => smaller treewidth.

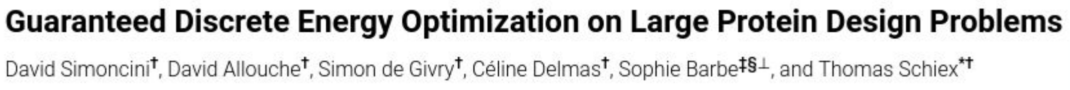


# Weaknesses & Strengths

- Not good for very large domains (time, scheduling)
- Not so good for random problems
- Optimization>feasibility (use SAT/ILP/CP if natural)

- Loves problem with a majority of functions over few (<=3) variables
- Useful when ‘small’ treewidth, or submodularity is present
- Unexpected efficiency on physics-based Computational Protein Design


algorithms
Guaranteed Diversity and Optimality in Cost Function Network Based Computational Protein Design Methods
Volume 14 · Issue 6 | June 2021
MDPI
mdpi.com/journal/algorithms
ISSN 1999-4893



**Guaranteed Discrete Energy Optimization on Large Protein Design Problems**

David Simoncini[†], David Allouche[†], Simon de Givry[†], Céline Delmas[†], Sophie Barbe[‡§⊥], and Thomas Schiex[*†]

JCTC Journal of Chemical Theory and Computation

IUCrJ
www.iucrj.org
Volume 6 | Part 1 | 1 January 2019
ISSN 2052-2525
International Union of CRYSTALLOGRAPHY

# Learning models from solutions (self-supervised, stochastic interpretation)

- From a set of ‘good’ solutions
  - Approximate log-likelihood with L1/L2 regularisation (sparsistent, sufficient statistics)
  - Relies on convex optimisation (ADMM)
  - CFN-learn numpy-based package, separate from toulbar2.

  Learn customer preferences from configurations (Renault)
  Learn how to play Sudoku (9,000 solved grids)

  https://github.com/toulbar2/CFN-learn

- From a set of good solutions with associated information (~~supervision~~):
  - Deep learning based (in: informations, out: a CFN)
  - Emmental-PLL loss (improves Besag consistent pseudo-loglikelihood - IJCAI’23)
  - Emmental-PLL torch-based Package, separate from toulbar2 - limitations

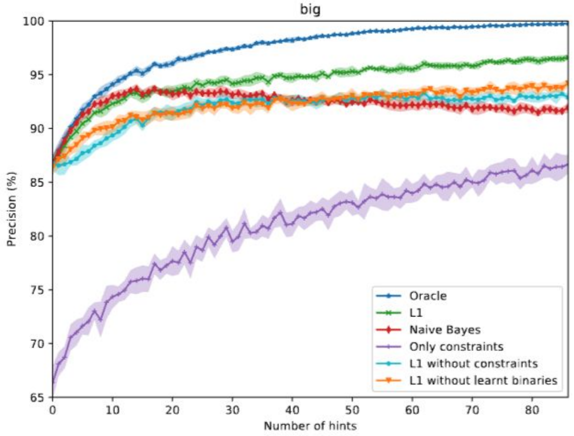
  Learn customer preferences from configurations (Renault) given age, SCP, gender
  Learn how to play Sudoku from the grid geometry (200 solved grids, image input)

  https://forgemia.inra.fr/marianne.defresne/emmental-pll

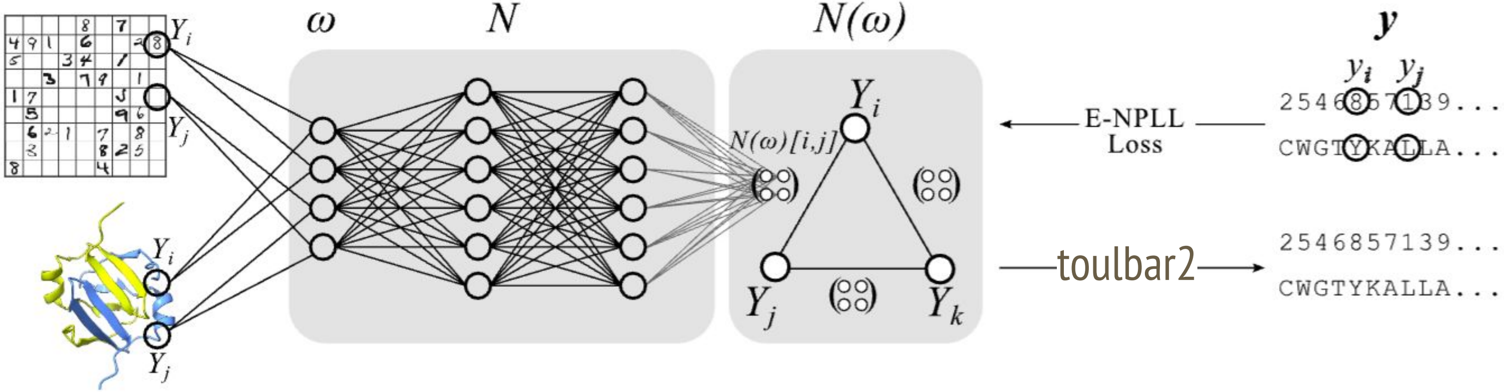
# Learning preferences from configurations

Renault utility van with combinatorial options:

- 68 variables, 324 values, 332 constraints (12 vars), 8,337 configurations
- Up to 24, 566, 537, 954, 855, 758, 069, 760 different vehicles
- Learning user preferences from passed valid configurations
- 10-fold cross validation



# Learning how to design proteins *(or play Sudoku)*



toulbar2

The learned representation of *p(y|ω)* can be constrained or biased arbitrarily w/o retraining.

# Learning how to design proteins

## Self-assembling complex

- Complex symmetry
- Specific interactions

## Ancestral protein

- Symmetry
- Simple chemistry

## Nanobodies

- DDPM (loop generation)
- Affinity & specificity