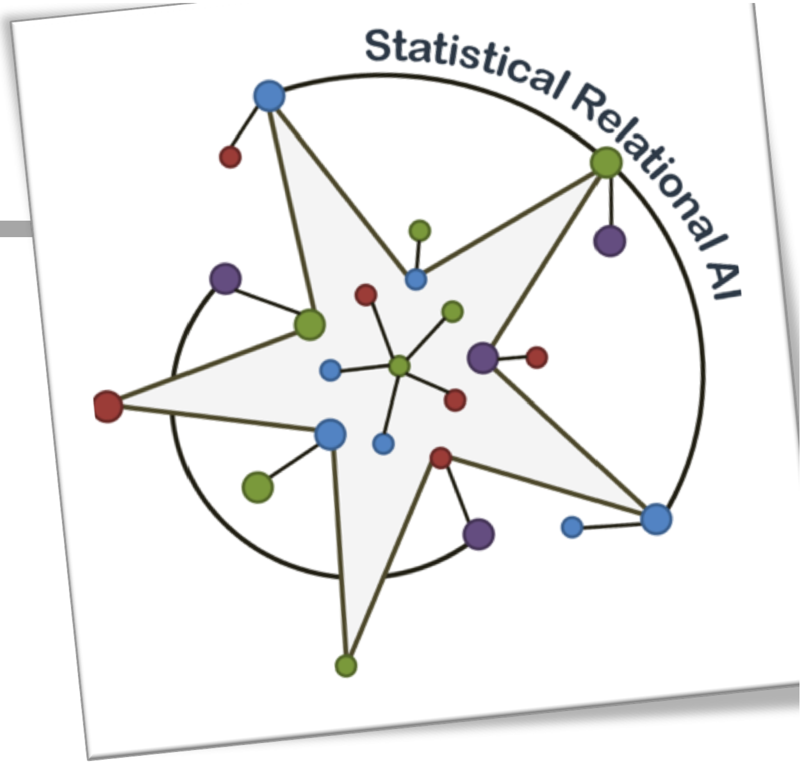
Approximate Lifted Inference on Relational Models

Statistical Relational AI



Tutorial at KI-2018

Tanya Braun, Universität zu Lübeck

Kristian Kersting, Technische Universität Darmstadt

Ralf Möller, Universität zu Lübeck

UNIVERSITÄT ZU LÜBECK

TECHNISCHE UNIVERSITÄT DARMSTADT

Lifted Approximate Inference

One way to get an approximate lifted inference approach is to replace "conditioning" by "sampling" in recursive conditioning approaches [see e.g. Gogate, Jha, Venugopal NIPS'12; Venugopal, Sarkhel, Gogate AAAI'15]

Lifted Belief Propagation [Jaimovich-UAI07, Singla-AAAI08, Kersting-UAI09]

Lifted Bisimulation/Mini-buckets [Sen-VLDB08, Sen-UAI09]

Lifted Importance Sampling [Gogate-UAI11, Gogate-AAAI12]

Lifted Relax, Compensate & Recover (Generalized BP) [VdB-UAI12]

Lifted MCMC [Niepert-UAI12, Niepert-AAAI13, Venugopal-NIPS12]

Lifted Variational Inference [Choi-UAI12, Bui-StarAI12]

Lifted MAP-LP [Mladenov-AISTATS14, Apsel-AAAI14] and many more …

Lifted Approximate Inference

One way to get an approximate lifted inference approach is to replace "conditioning" by "sampling" in recursive conditioning approaches [see e.g. Gogate, Jha, Venugopal NIPS'12; Venugopal, Sarkhel, Gogate AAAI'15]

- Here, we want to take an algebraic, group-theoretical view on approximate lifted inference
- This provides a general understanding across different families of inference algorithms.
- To do so, we start by lifting (loopy) belief propagation

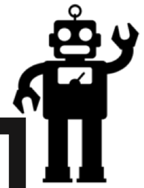
[Singla, Domingos AAAI'08; Kersting, Ahmadi, Natarajan UAI'09; Ahmadi, Kersting, Mladenov, Natarajan MLJ'13]

Lifted Loopy Belief Propagation Exploiting computational symmetries

Big
Model

If exchanging two variables preserves optimality, group them together

automatically
compressed



Small
Model

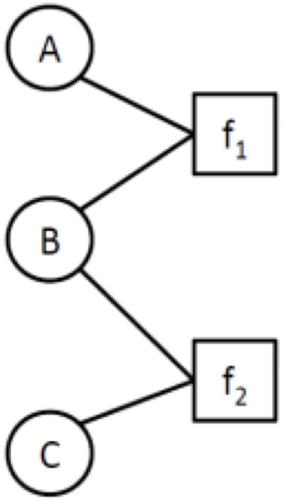
Run
Loopy Belief Propagation

Run a modified
Loopy Belief Propagation

What are symmetries in (loopy) belief propagation?

Compression: Pass the colors around*

*can also be done at the „lifted“, i.e., relational level

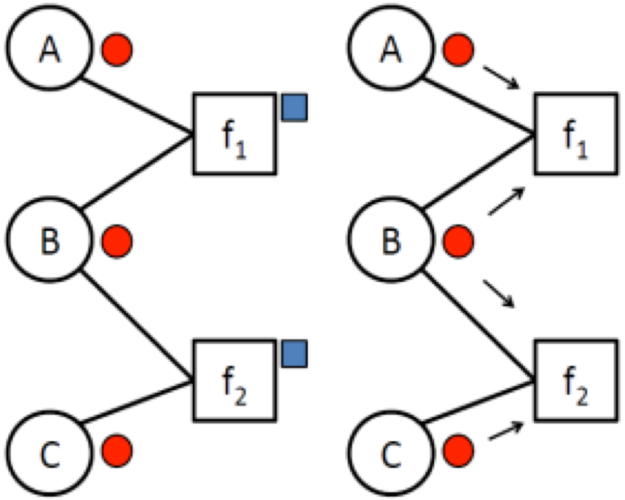


- **Color nodes according to the evidence you have**
  - No evidence, say **red**
  - State „one“, say **brown**
  - State „two“, say **orange**
  - ...
- **Color factors distinctively according to their equivalences** For instance, assuming f_1 and f_2 to be identical and B appears at the second position within both, say **blue**

[Singla, Domingos AAAI'08; Kersting, Ahmadi, Natarajan UAI'09; Ahmadi, Kersting, Mladenov, Natarajan MLJ'13]

Compression: Pass the colors around*

*can also be done at the „lifted", i.e., relational level

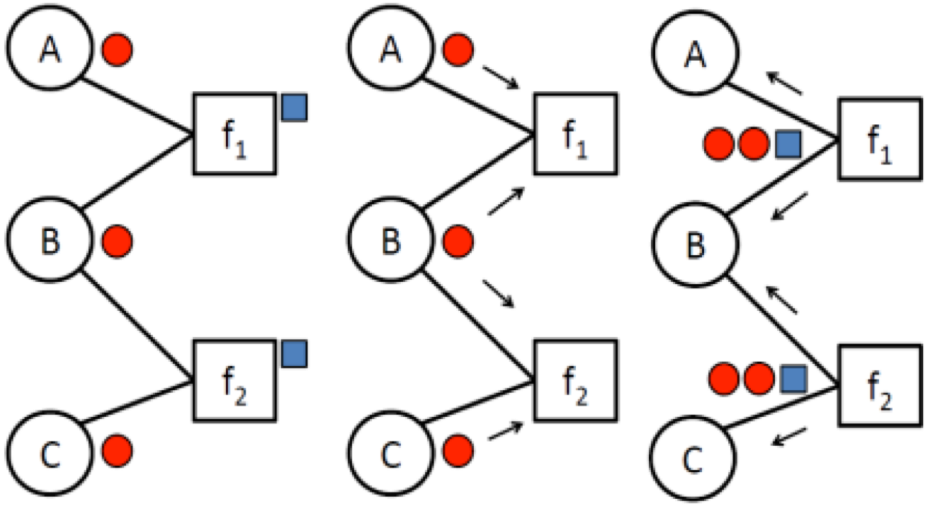


1. Each factor collects the colors of its neighboring nodes

[Singla, Domingos AAAI'08; Kersting, Ahmadi, Natarajan UAI'09; Ahmadi, Kersting, Mladenov, Natarajan MLJ'13]

Compression: Pass the colors around*

*can also be done at the „lifted", i.e., relational level

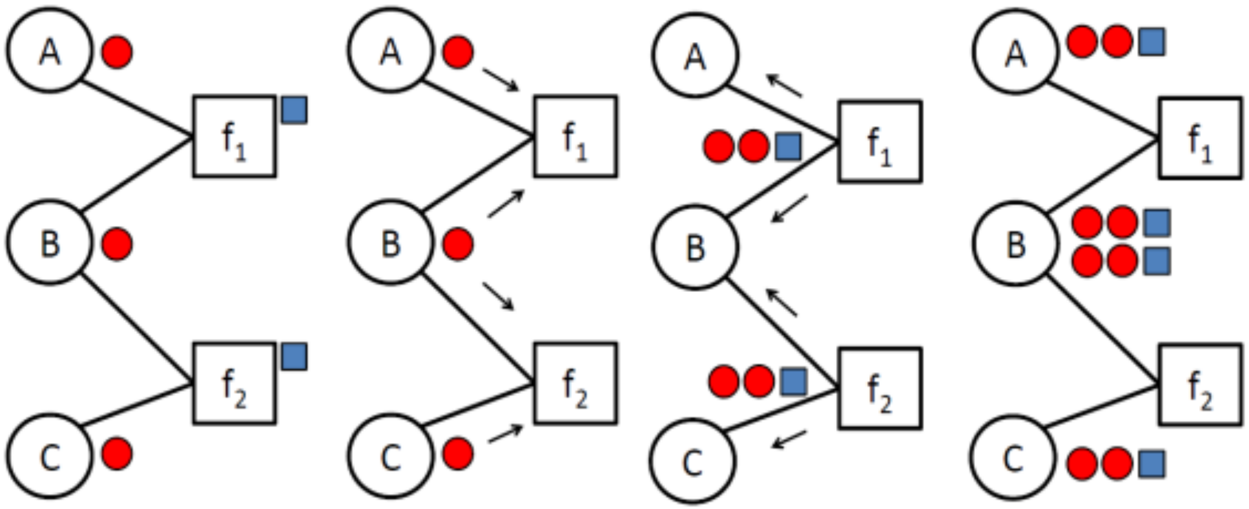


1. Each factor collects the colors of its neighboring nodes
2. Each factor „signs" ist color signature with its own color

[Singla, Domingos AAAI'08; Kersting, Ahmadi, Natarajan UAI'09; Ahmadi, Kersting, Mladenov, Natarajan MLJ'13]

Compression: Pass the colors around*

*can also be done at the „lifted“, i.e., relational level

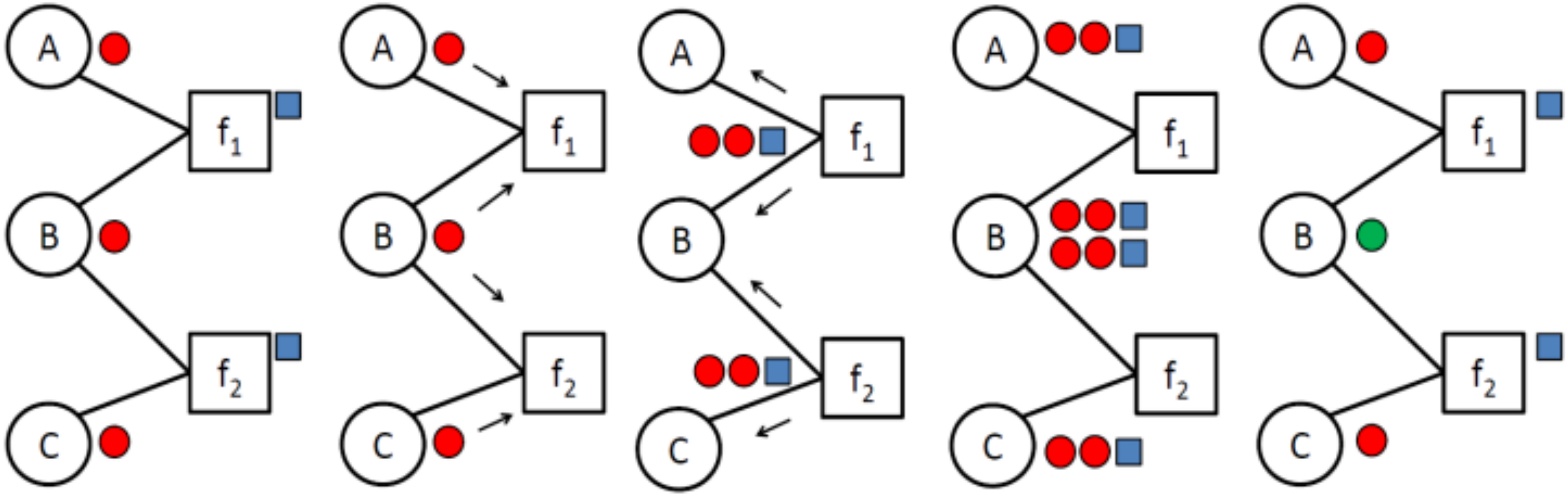


1. Each factor collects the colors of its neighboring nodes
2. Each factor „signs“ ist color signature with its own color
3. Each node collects the signatures of its neighboring factors

[Singla, Domingos AAAI'08; Kersting, Ahmadi, Natarajan UAI'09; Ahmadi, Kersting, Mladenov, Natarajan MLJ'13]

Compression: Pass the colors around*

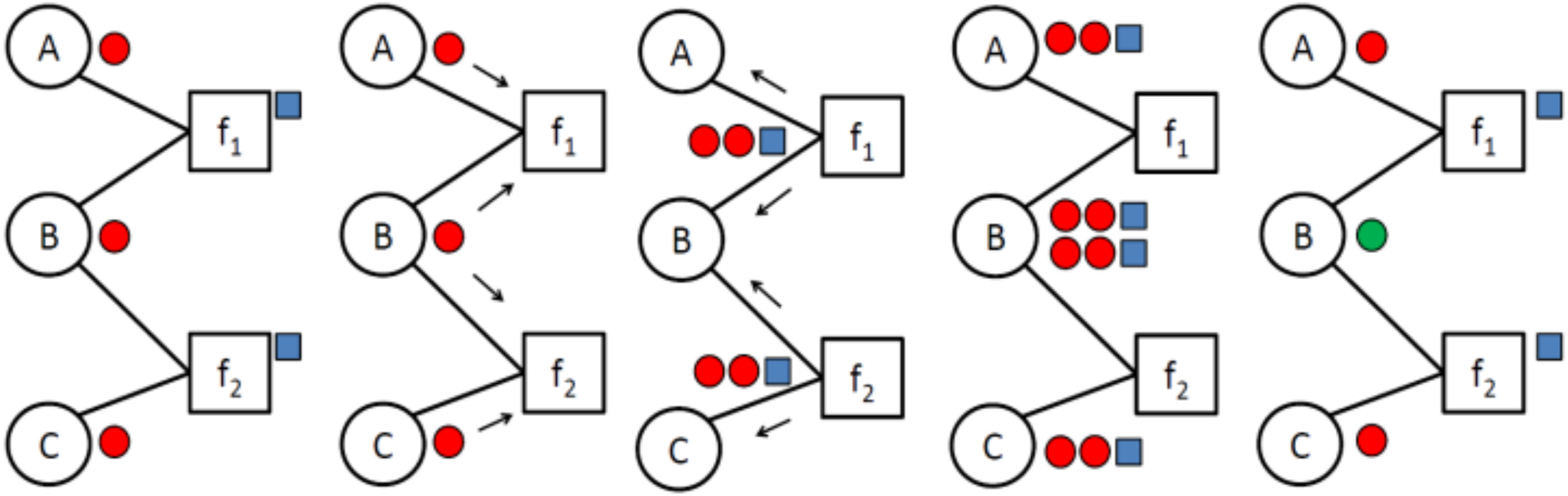
*can also be done at the „lifted", i.e., relational level



1. Each factor collects the colors of its neighboring nodes
2. Each factor „signs" ist color signature with its own color
3. Each node collects the signatures of its neighboring factors
4. Nodes are recolored according to the collected signatures

Compression: Pass the colors around*

*can also be done at the „lifted“, i.e., relational level



1. Each factor collects the colors of its neighboring nodes
2. Each factor „signs“ ist color signature with its own color
3. Each node collects the signatures of its neighboring factors
4. Nodes are recolored according to the collected signatures
5. If no new color is created stop, otherwise go back to 1

Compression can considerably speed up inference and training

Probabilistic inference using lifted (loopy) belief propagation

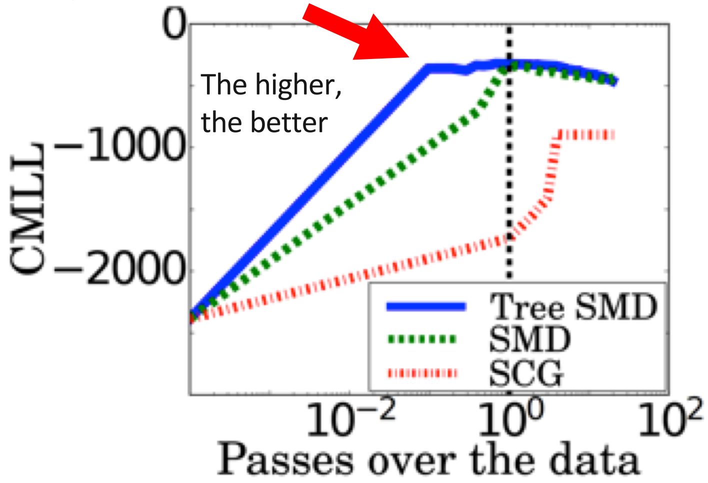
Domain	Time (in seconds) The lower, the better						No. of (Super) Features	
	Construction		BP		Total			
	Ground	Lifted	Ground	Lifted	Ground	Lifted	Ground	Lifted
Cora	263.1	1173.3	12368.4	3997.7	12631.6	5171.1	2078629	295468
UW-CSE	6.9	22.1	1015.8	602.5	1022.8	624.7	217665	86459
Friends & Smokers	38.8	89.7	10702.2	4.4	10741.0	94.2	1900905	58

114x faster

Parameter training using a lifted stochastic gradient

CORA entity resolution

converges before data has been seen once

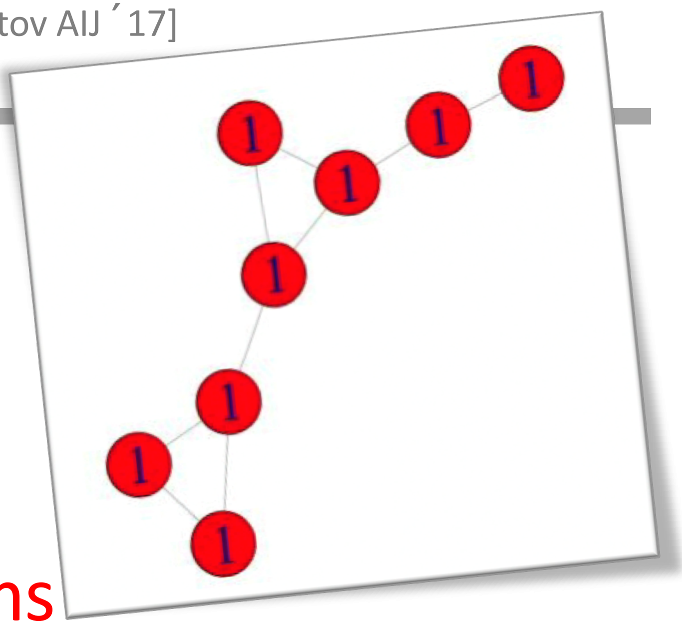


State-of-the-art

What is going on algebraically?

Can we generalize this to other ML approaches?

[Mladenov, Ahmadi, Kersting AISTATS´12, Grohe, Kersting, Mladenov, Selman ESA´14, Mladenov, Globerson, Kersting UAI ´14, AISTATS ´14, Mladenov, Kersting UAI ´15, Kersting, Mladenov, Tokmatov AIJ ´17]



- AKA Naive Vertex Classification
- Basic subroutine for GI testing
- Computes LP-relaxations of GA-ILP, aka. fractional automorphisms
- Quasi-linear running time O((n+m)log(n)) when using asynchronous updates [Berkholz, Bonsma, Grohe ESA´13]
- Part of graph tool SAUCY [See e.g. Darga, Sakallah, Markov DAC´08]

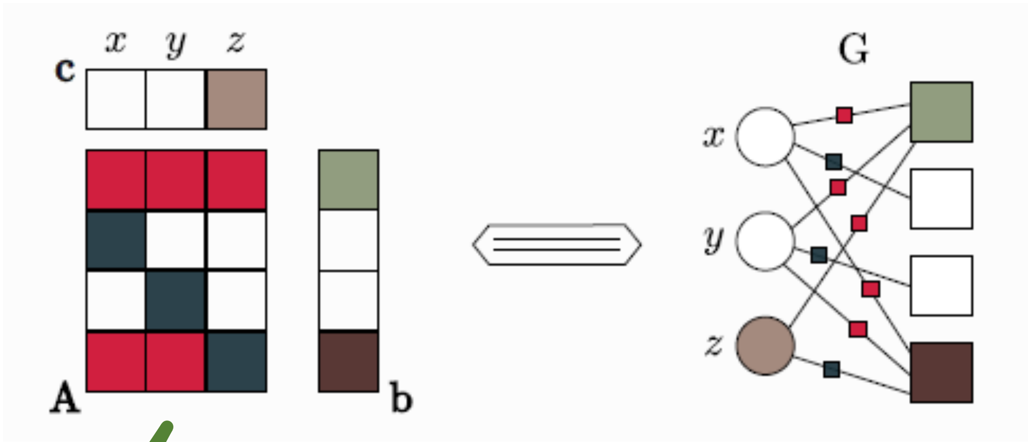
It turns out that color passing is well known in graph theory

(1) The Weisfeiler-Lehman Algorithm

[Mladenov, Ahmadi, Kersting AISTATS ´12, Grohe, Kersting, Mladenov, Selman ESA ´14, Mladenov, Globerson, Kersting UAI ´14, AISTATS ´14, Mladenov, Kersting UAI ´15, Kersting, Mladenov, Tokmatov AIJ ´17]

$$\max_{[x,y,z]^T \in \mathbb{R}^3} \quad 0x + 0y + 1z$$

$$\text{s.t.} \quad \begin{bmatrix} 1 & 1 & 1 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \\ 1 & 1 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \leq \begin{bmatrix} 1 \\ 0 \\ 0 \\ -1 \end{bmatrix}$$

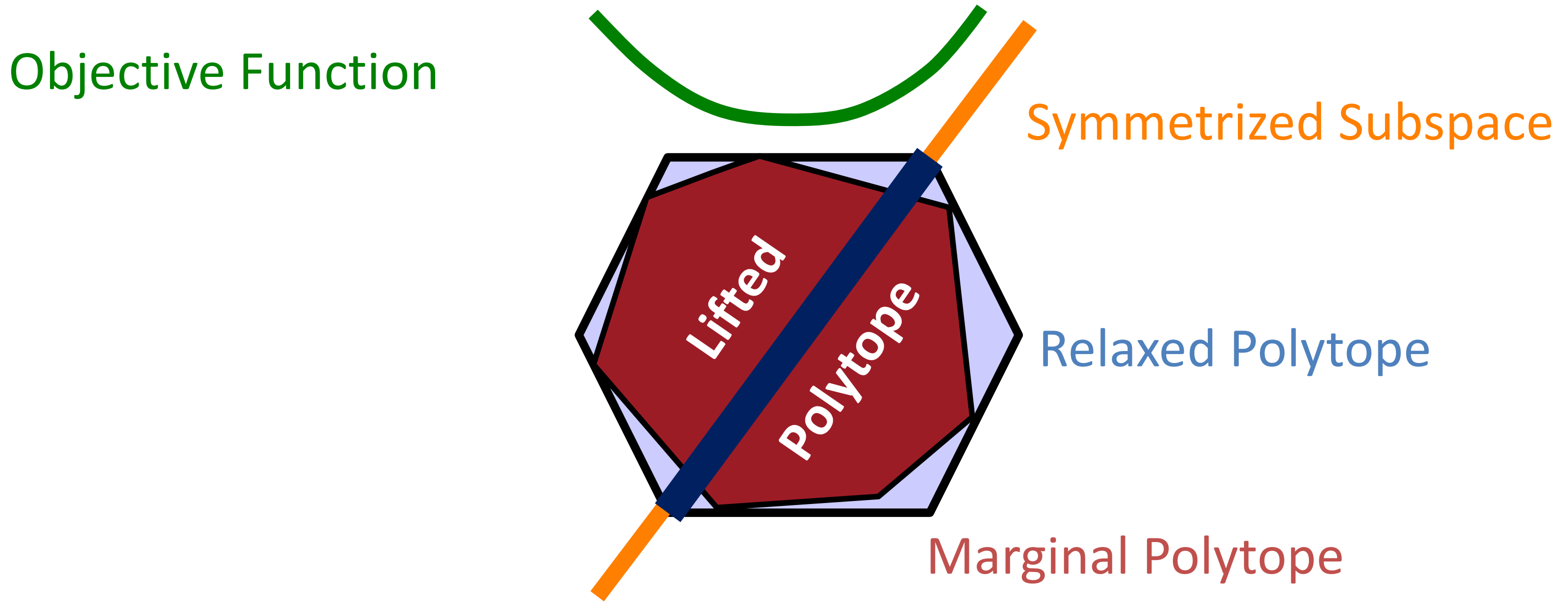


$$\mathbf{X}_{\color{blue}{Q}} \mathbf{A} = \mathbf{A} \mathbf{X}_{\color{green}{P}}$$

(2) Realize that WL computes (fractional) automorphisms of mathmetical programs

[Mladenov, Ahmadi, Kersting AISTATS ´12, Grohe, Kersting, Mladenov, Selman ESA ´14, Mladenov, Globerson, Kersting UAI ´14, AISTATS ´14, Mladenov, Kersting UAI ´15, Kersting, Mladenov, Tokmatov AIJ ´17]

$$\hat{\mathbf{x}} \in \arg\max_{\mathbf{x}\in\mathcal{X}^N} \left\{ \sum_{s\in V} \theta_s(x_s) + \sum_{(s,t)\in E} \theta_{st}(x_s, x_t) \right\}$$

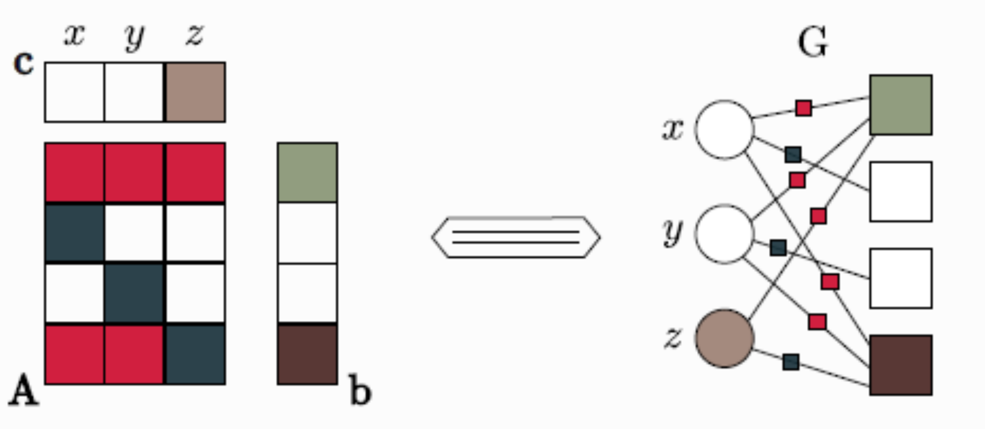


(3) Apply this to probabilistic inference

Lifted Mathematical Programming
Exploiting computational symmetries

[Mladenov, Ahmadi, Kersting AISTATS ´12, Grohe, Kersting, Mladenov, Selman ESA ´14, Kersting, Mladenov, Tokmatov AIJ ´17]

$$\max_{[x,y,z]^T \in \mathbb{R}^3} \quad 0x + 0y + 1z$$
$$\text{s.t.} \quad \begin{bmatrix} 1 & 1 & 1 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \\ 1 & 1 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \leq \begin{bmatrix} 1 \\ 0 \\ 0 \\ -1 \end{bmatrix}$$



View the mathematical program as a colored graph

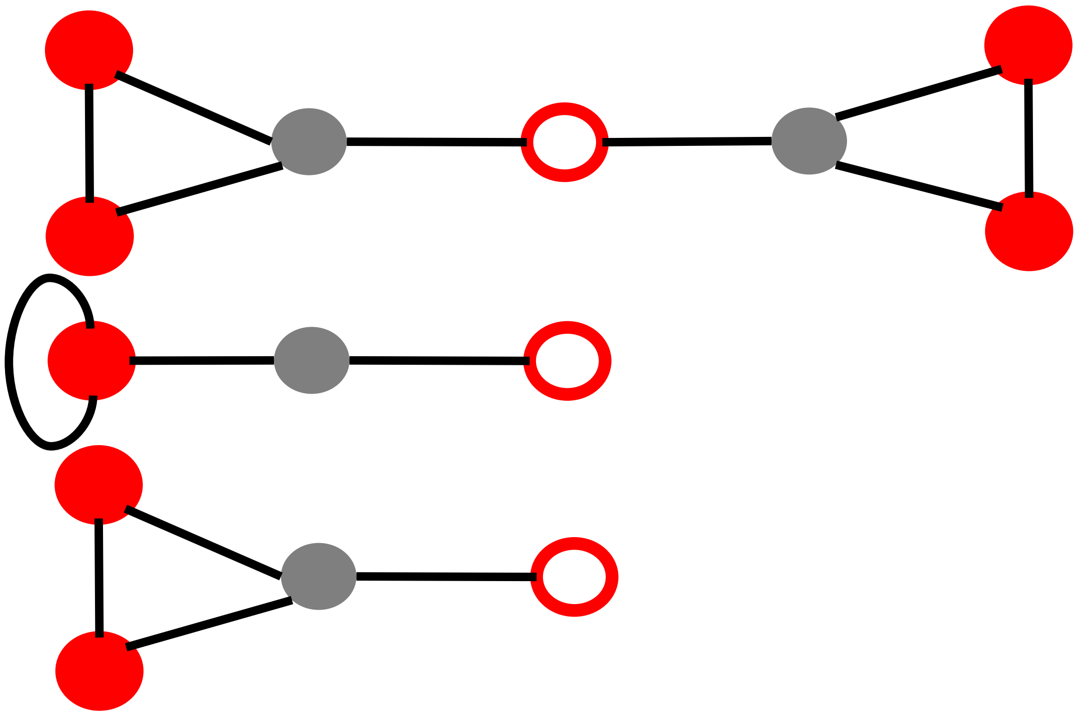
Reduce the mathematical program (MP) by running Weisfeiler-Lehman on the MP-Graph

Solve the reduce MP using <u>any</u> solver

Any Solver? Well, you can lifted optimization by reparametrization

Attention: For special-purpose solvers such as message-passing (via coordinate descent) for probabilistic inference we may have to reparameterize the lifted model

[Mladenov, Globerson, Kersting UAI 2014; Mladnov, Kersting UAI 2015]

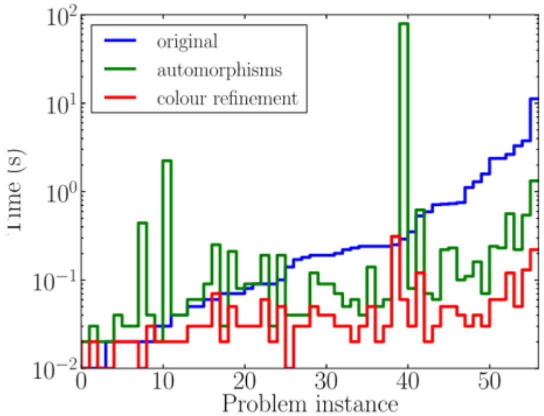


lifting

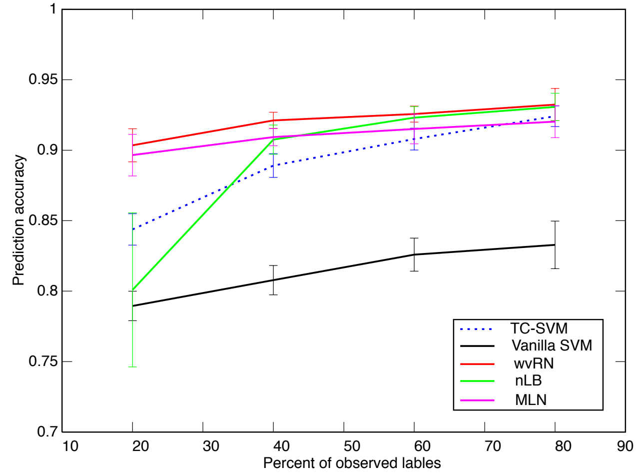
refine

[Grohe, Kersting, Mladenov, Selman ESA´14, Kersting, Mladenov, Tokmatov AIJ´17]

Lifted Linear Programming

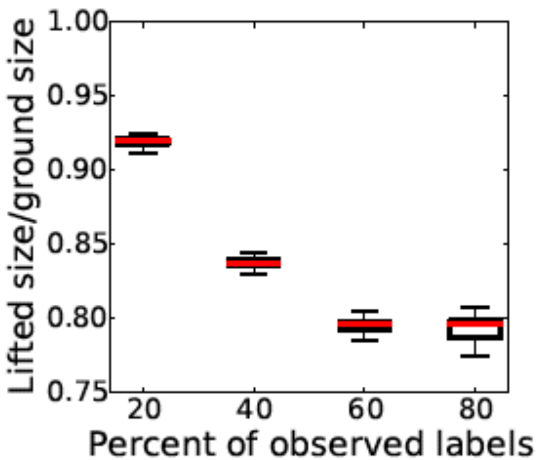


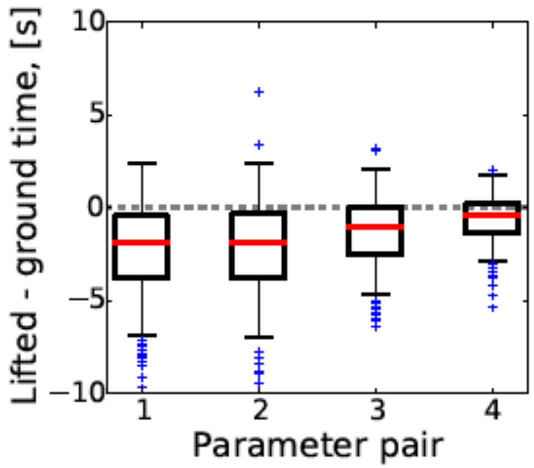
Margout‘s ILPs
with symmetries
(relaxed)

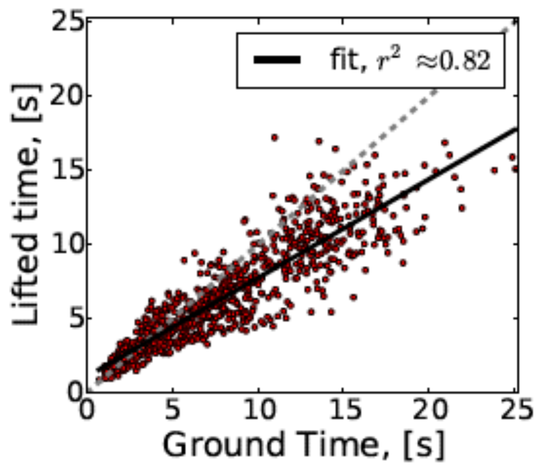


Collective Classification

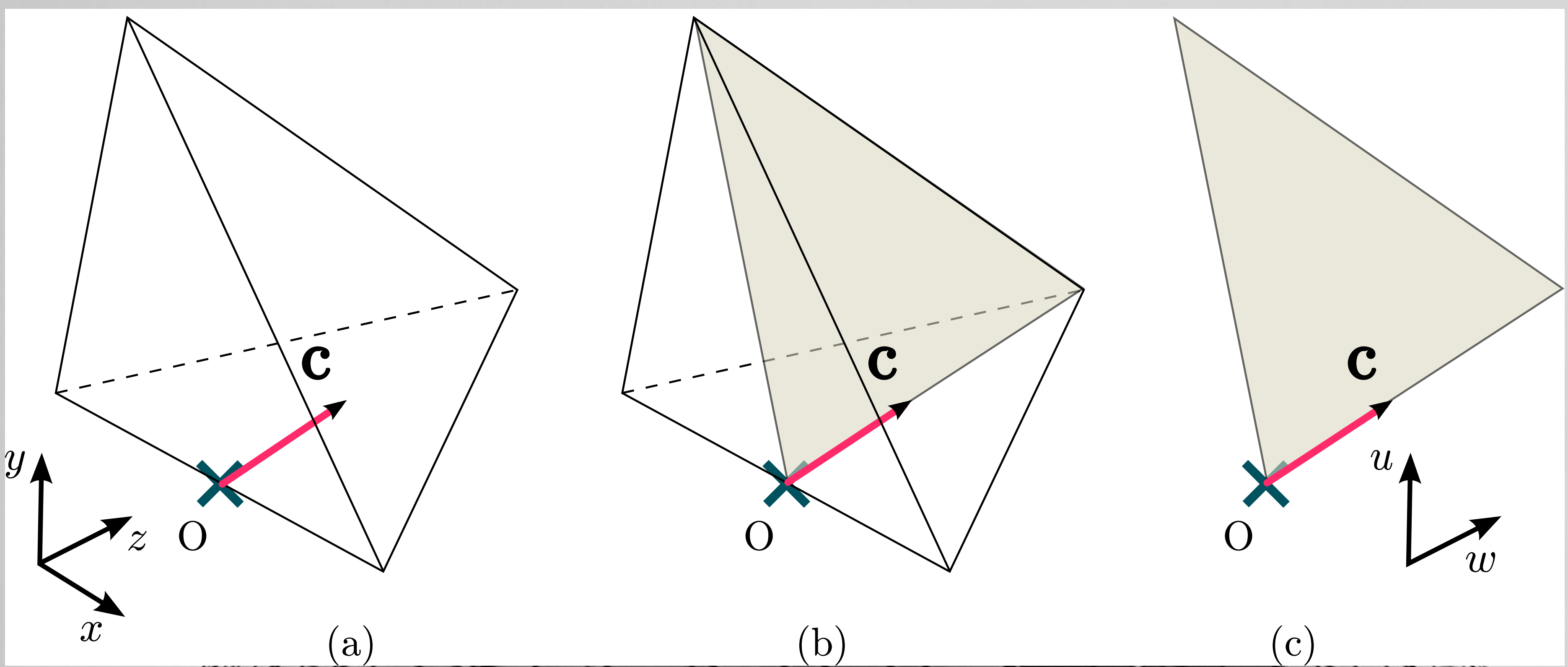
Cora (most common vs. rest)







The more observed the more lifting. Faster end-to-end even despite Gurobi‘s fast pre-solving heuristics



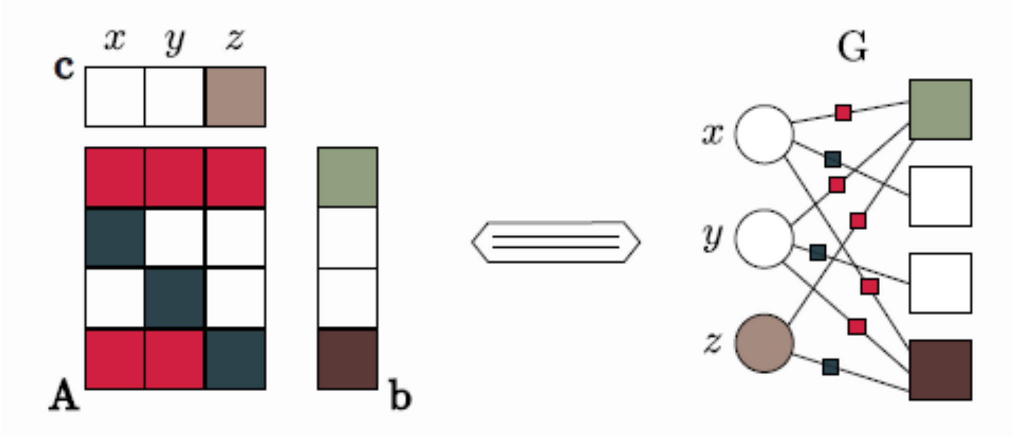
Feasible region of LP and the objective vectors

Span of the fractional auto-morphism of the LP

Projections of the feasible region onto the span of the fractional auto-morphism

Why does this work?

Compute Equitable Partition (EP) of the LP using WL



$$\mathcal{P} = \underbrace{\{P_1, \ldots, P_p;}_{\text{Partition of LP variables}} \underbrace{Q_1, \ldots, Q_q\}}_{\text{Partition of LP constraints}}$$

Intuitively, we group together variables resp. constraints that interact in the very same way in the LP.

Fractional Automorphisms of LPs

The EP induces a fractional automorphism of the coefficient matrix **A**

$$\mathbf{X}_Q \mathbf{A} = \mathbf{A} \mathbf{X}_P$$

where $\mathbf{X}_Q$ and $\mathbf{X}_p$ are doubly-stochastic matrixes (relaxed form of automorphism)

$$(\mathbf{X}_P)_{ij} = \begin{cases} 1/|P| & \text{if both vertices } i,\ j \text{ are in the same } P, \\ 0 & \text{otherwise.} \end{cases}$$

$$(\mathbf{X}_Q)_{ij} = \begin{cases} 1/|Q| & \text{if both vertices } i,\ j \text{ are in the same } Q, \\ 0 & \text{otherwise} \end{cases}$$

Fractional Automorphisms Preserve Solutions

If $\mathbf{x}$ is feasible, then $\mathbf{X}_p\mathbf{x}$ is feasible, too.

By induction, one can show that left-multiplying with a double-stochastic matrix preserves directions of inequalities. Hence,

$$\mathbf{A}\mathbf{x} \leq \mathbf{b} \Rightarrow \mathbf{X}_Q\mathbf{A}\mathbf{x} \leq \mathbf{X}_Q\mathbf{b} \Leftrightarrow \mathbf{A}\mathbf{X}_P\mathbf{x} \leq \mathbf{b}$$

Fractional Automorphisms Preserve Solutions

If $\mathbf{x}^*$ is optimal, then $\mathbf{X}_p\mathbf{x}^*$ is optimal, too.

Since by construnction $\mathbf{c}^T\mathbf{X}_P = \mathbf{c}^T$ and hence $\mathbf{c}^T(\mathbf{X}_P\mathbf{x}) = \mathbf{c}^T\mathbf{x}$

What have we established so far?

Instead of considering the original LP

$$(\mathbf{A}, \mathbf{b}, \mathbf{c})$$

It is sufficient to consider

$$(\mathbf{A}\mathbf{X}_P, \mathbf{b}, {\mathbf{X}_P}^T\mathbf{c})$$

i.e. we “average” parts of the polytope.

But why is this dimensionality reduction?

Dimensionality Reduction

The doubly-stochastic matrix $\mathbf{X}_P$ can be written as

$$\mathbf{X}_P = \mathbf{B}\mathbf{B}^T$$

$$\mathbf{B}_{iP} = \begin{cases} \frac{1}{\sqrt{|P|}} & \text{if vertex } i \text{ belongs to part } P, \\ 0 & \text{otherwise.} \end{cases}$$

Since the column space of B is equivalent to the span of $\mathbf{X}_P$, it is actually sufficient to consider only

$$(\mathbf{A}\mathbf{B}_P, \mathbf{b}, \mathbf{B}_P^T\mathbf{c})$$

This is of reduced size, and actually we can also drop any constraints that becomes identical

$$\mathbf{X}_Q \mathbf{A} = \mathbf{A} \mathbf{X}_P$$

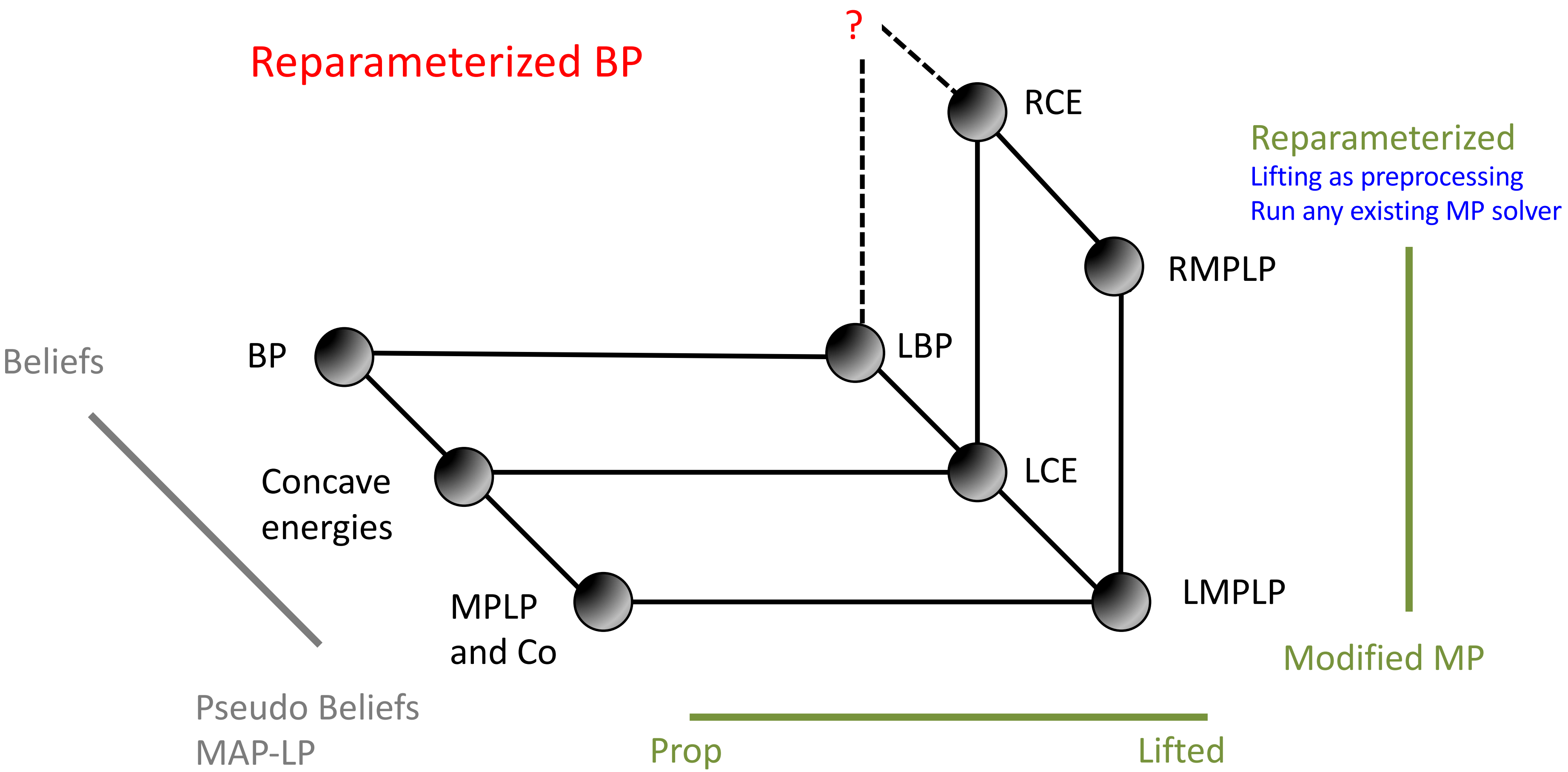
Fractional automorphisms provide an algebraic tool to study lifted inference

Actually, there is a whole body of work on (fractional) automorphisms for probabilistic inference, see the book, and we have focused here on the arguably simplest view.

This has resulted in an important insight ...

Lifted inference =

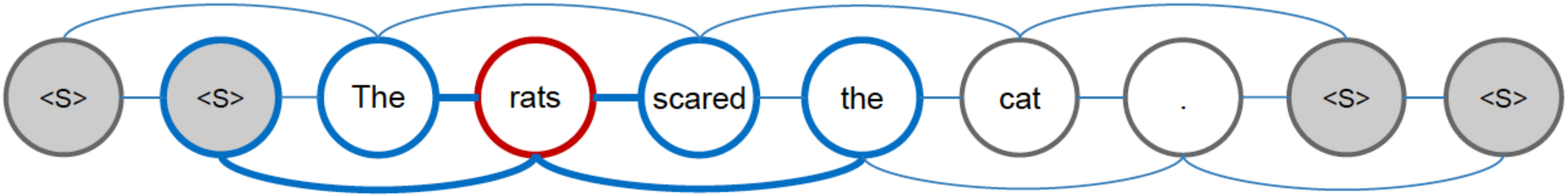
Inference in a smaller, reparameterized model



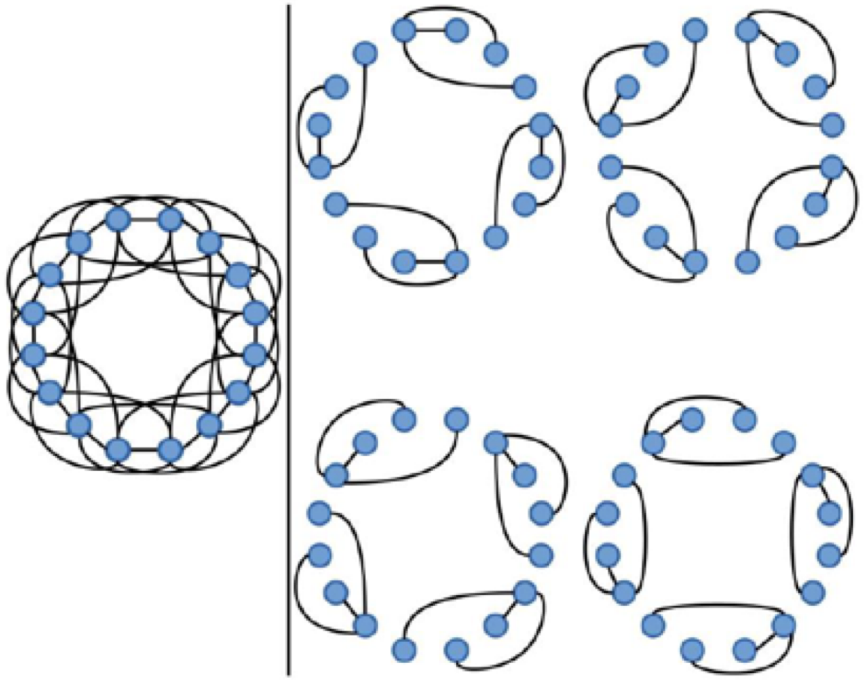
This can also speed up learning

[Jernite, Rush, Sontag ICML 2015]

Lifted Learning of MRF Language Models



- Word distribution for all sentences together
- Dependencies on size K context per sentence
- Exploit symmetries



And extends lifting to statistical ML

Lifted Convex Quadratic Programs

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}\in\mathcal{D}} J(\boldsymbol{x})$$
$$J(\boldsymbol{x}) = \boldsymbol{x}^T Q\boldsymbol{x} + \boldsymbol{c}^T\boldsymbol{x}$$
$$\mathcal{D} = \{\boldsymbol{x} : A\boldsymbol{x} \leqslant \boldsymbol{b}\}$$

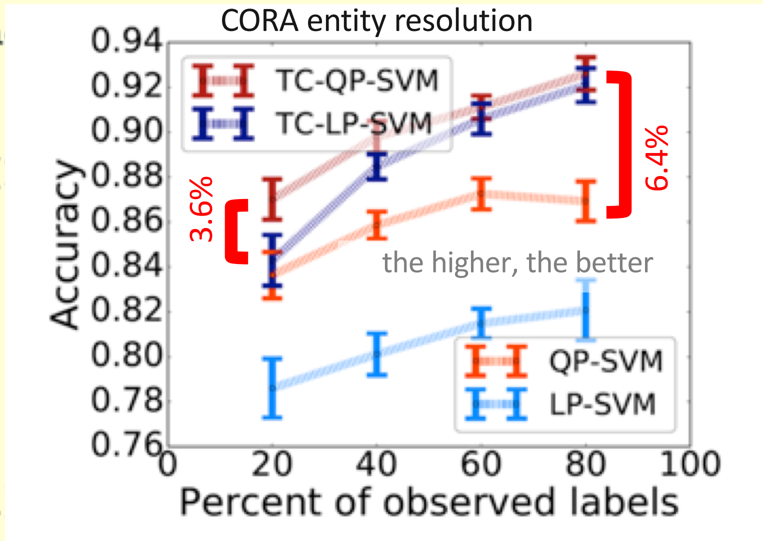
```
#QUADRATIC OBJECTIVE
minimize: sum{J in feature(I,J)} weight(J)**2 + c1 * sla

#labeled examples should be on the correct side
subject to forall {I in labeled(I)}: labeled(I)*predict(

#slack
subjec

#TRANSDUCTIVE PART
#cited instances should have the same labels.
subject to forall {I1, I2 in linked(I1, I2)}: labeled(I1                          ;
subject to forall {I1, I2 in linked(I1, I2)}: coslack(I1, I2) >= 0; #coslacks are positive
```

On par with state-of-the-art by just four lines of code

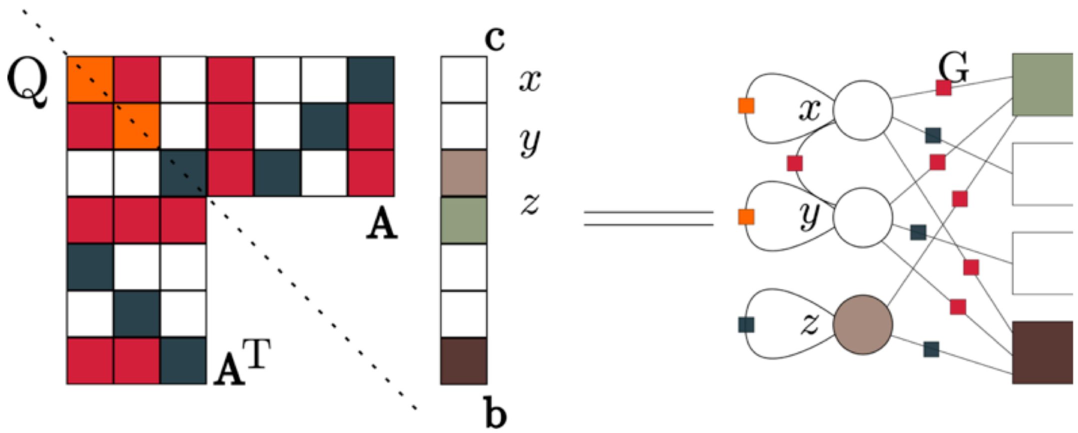


Papers that cite each other should be on the same side of the hyperplane

Reduce the QP by running Weisfeiler-Lehman on the QP-Graph

$$\max_{[x,y,z]^T\in\mathbb{R}^3} \quad 0x + 0y + 1z - 1z^2 - 2x^2 - 2y^2 + 1xy + 1yx$$

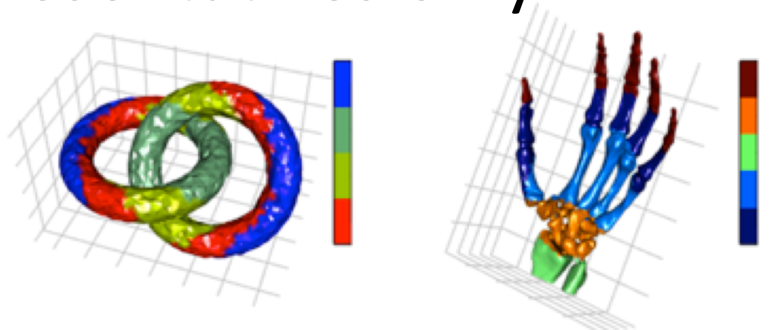
$$\text{s.t.} \quad \begin{bmatrix} 1 & 1 & 1 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \\ 1 & 1 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \leq \begin{bmatrix} 1 \\ 0 \\ 0 \\ -1 \end{bmatrix}$$



[Mladenov, Kleinhans, Kersting AAAI ´17]

Approximately Lifted SVM:

Cluster data points via K-means using sorted distance vectors. Solve SVM on cluster representatives only

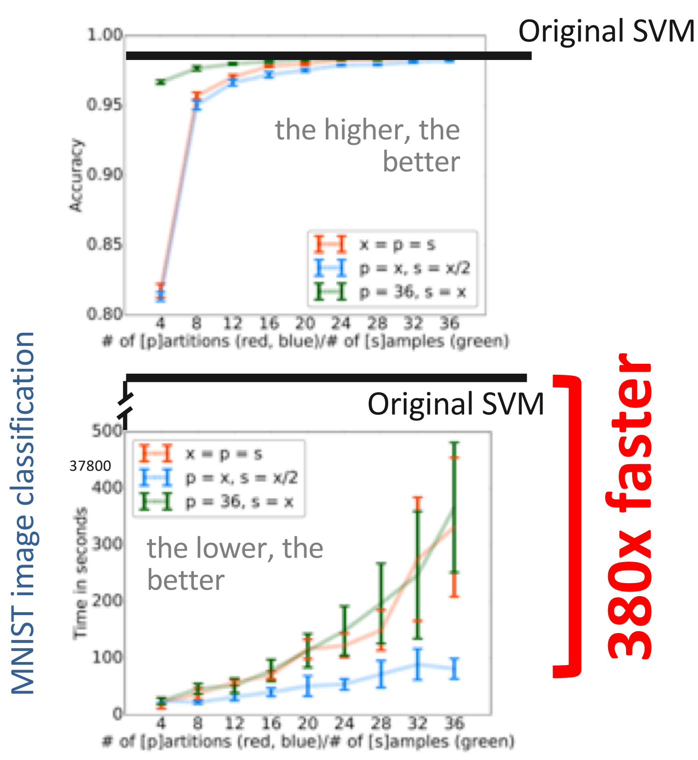


PAC-style general. bound:
the approximately lifted SVM will very likely have a small expected error rate if it has a small empirical loss over the original dataset.

Similar predictive performance but 47x faster

Symmetry-based Data Augmentation:

fractional autom. of label-preserving data transformations



(d) Learning time: Translation

Same should work for deep learning

Industrial Strength Solvers such as CPLEX and GUROBI

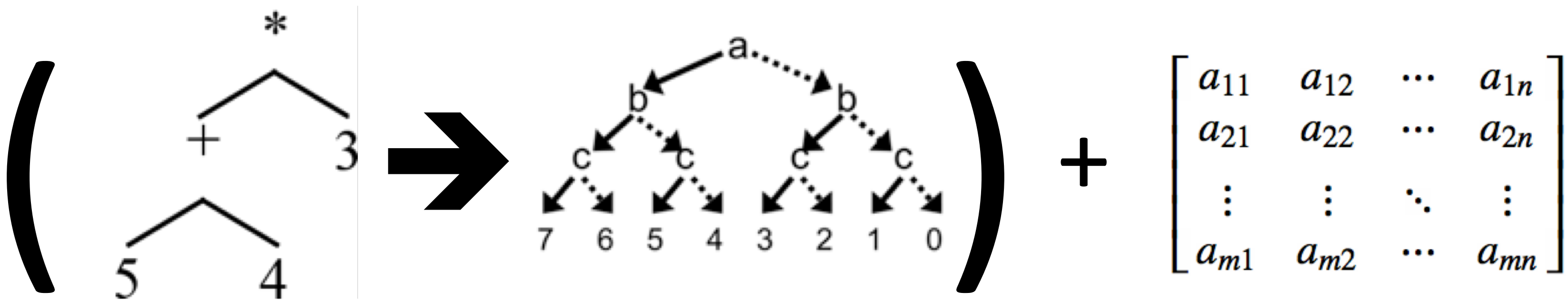
IBM®

GUROBI
OPTIMIZATION

[Mladenov, Belle, Kersting AAAI´17, Kolb, Mladenov, Sanner, Belle, Kersting IJCAI ECAI´18]

And, there are other “-02”, “-03”, … flags,
e.g symbolic-numerical interior point solvers

New field: Symbolic-numerical AI



Formulae parse trees

Algebraic Decision Diagrams

Matrix Free Optimization

Problem Statistics				Symbolic IPM		Ground IPM
name	#vars	#constr	$nnz(A)$	|ADD|	time[s]	time[s]
factory	131.072	688.128	4.000.000	1819	6899	**516**
factory0	524.288	2.752.510	15.510.000	1895	**6544**	7920
factory1	2.097.150	11.000.000	59.549.700	2406	**34749**	159730
factory2	4.194.300	22.020.100	119.099.000	2504	**36248**	$\geq$ 48hrs.

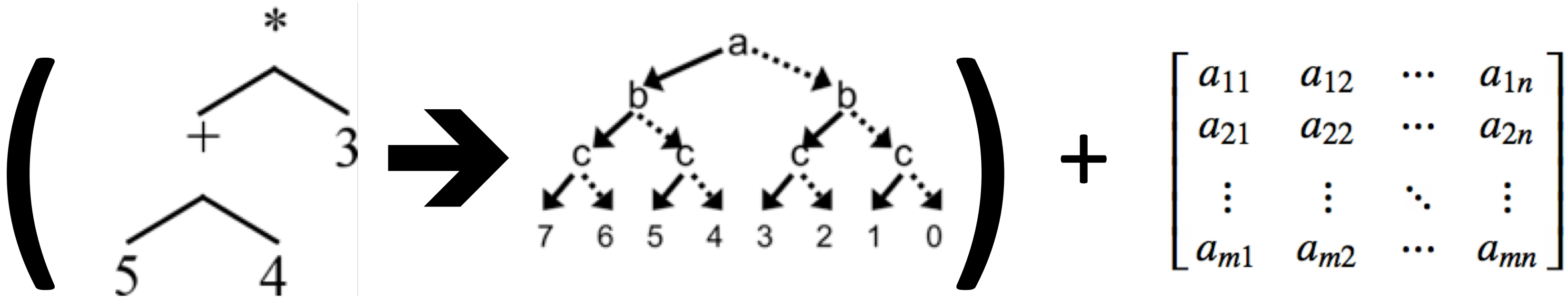
>4.8x faster

Applies to QPs but here illustrated on MDPs for a factory agent which must paint two objects and connect them. The objects must be smoothed, shaped and polished and possibly drilled before painting, each of which actions require a number of tools which are possibly available. Various painting and connection methods are represented, each having an effect on the quality of the job, and each requiring tools. Rewards (required quality) range from 0 to 10 and a discounting factor of 0. 9 was used used

[Mladenov, Belle, Kersting AAAI´17, Kolb, Mladenov, Sanner, Belle, Kersting IJCAI ECAI´18]

And, there are other "-02", "-03", … flags,
e.g symbolic-numerical interior point solvers

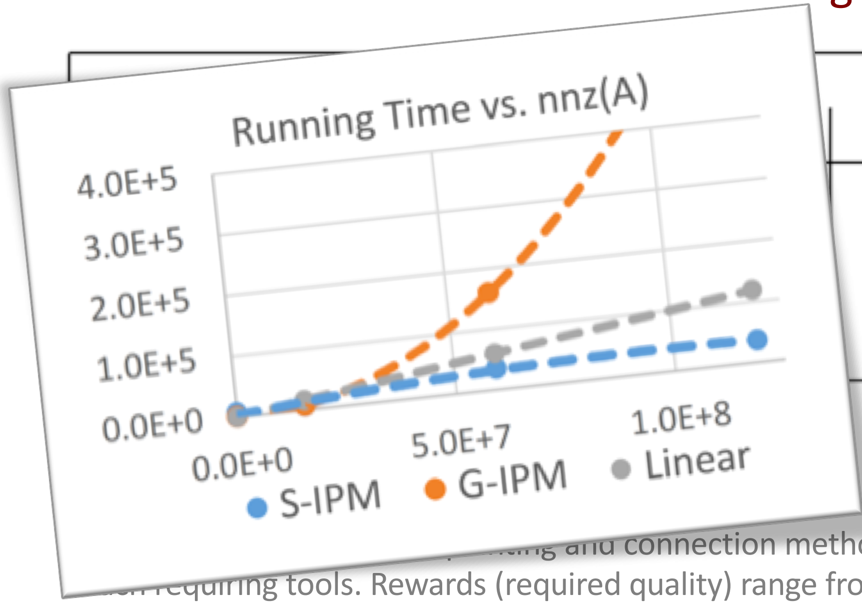
New field: Symbolic-numerical AI



Formulae parse trees

Algebraic Decision Diagrams

Matrix Free Optimization



All this opens the general machine learning toolbox for symbolic-numerical machines:
feature selection, least-squares regression, label propagation, ranking, collaborative filtering, community detection, deep learning, …

…ting and connection methods are represented, each having an effect on the quality of the job, and …requiring tools. Rewards (required quality) range from 0 to 10 and a discounting factor of 0. 9 was used used

Symmetries can also be exploited to speed up sampling

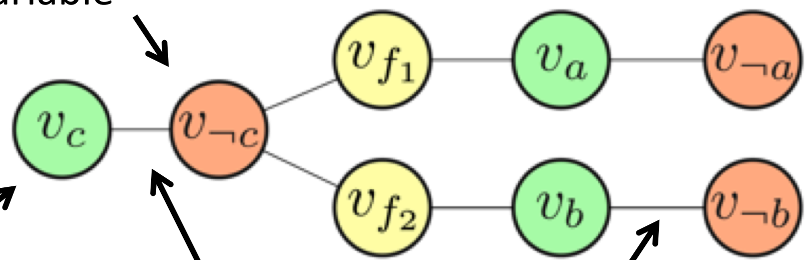
[Niepert UAI 2012, Van den Broeck, Niepert AAAI 2015]

Orbital Markov Chain Monte Carlo

true and false states have the same color, and all clauses/features that have the same weight

Feature/clause a variable participates in

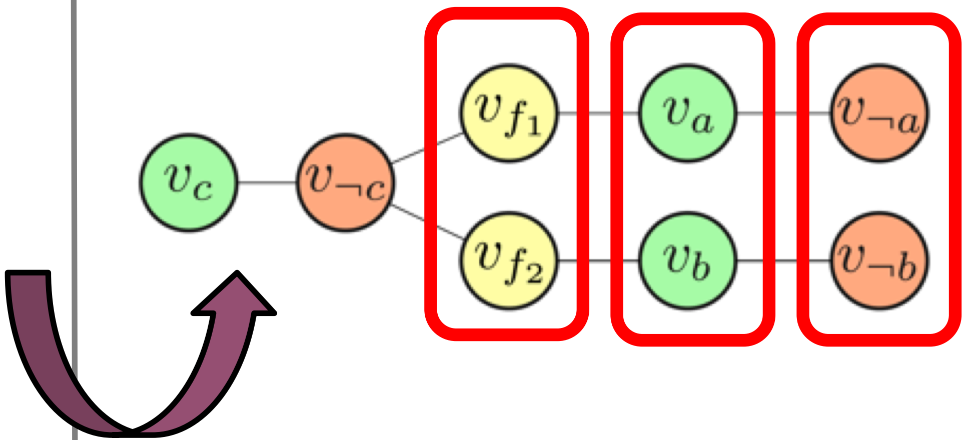
False state of variable



True state of a variable

States of a variable should not be in the same orbit

Symmetry classes of variables



Color-passing/Saucy

Jump between symmetric states uniformly

[Niepert UAI 2012, Van den Broeck, Niepert AAAI 2015]

Orbital MCMC Sampling

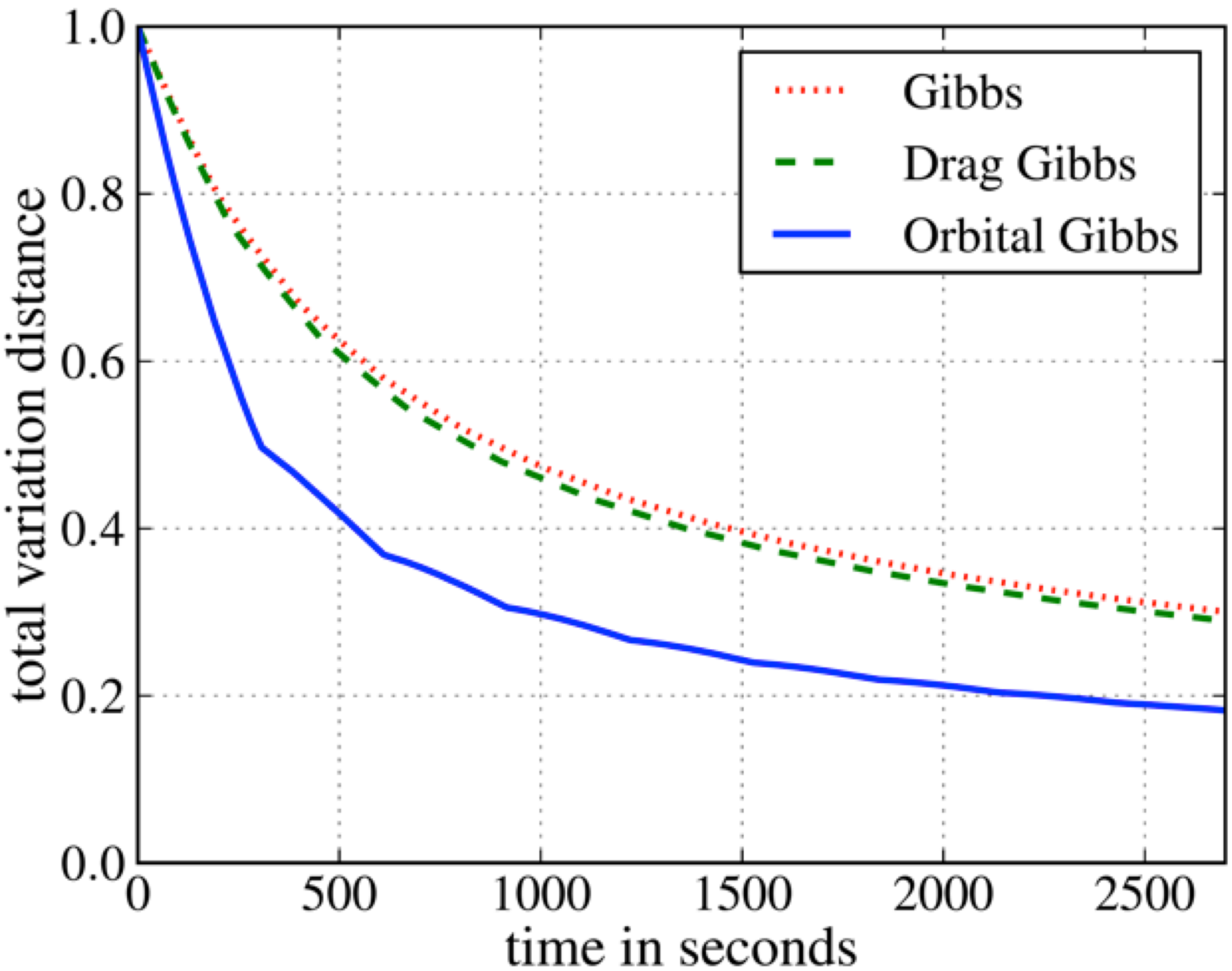
In each sampling iteration:

1. run a step of a traditional MCMC chain TM first and then

2. sample the state of M at the next time uniformly at random from the orbit of the state of the original chain TM at time t, i.e., select an equivalent state uniformly at random

[Niepert UAI 2012, Van den Broeck, Niepert AAAI 2015]

Orbital MCMC on a 6x6 Ising grid



[Niepert UAI 2012, Van den Broeck, Niepert AAAI 2015]

Lifted Metropolis-Hastings

Given an orbital Metropolis chain A:

- Given symmetry group G (approx. symmetries)
- Orbit $\mathbf{x}^G$ contains all states approx. symm. to $\mathbf{x}$
- In state $\mathbf{x}$:

Color-passing /Saucy

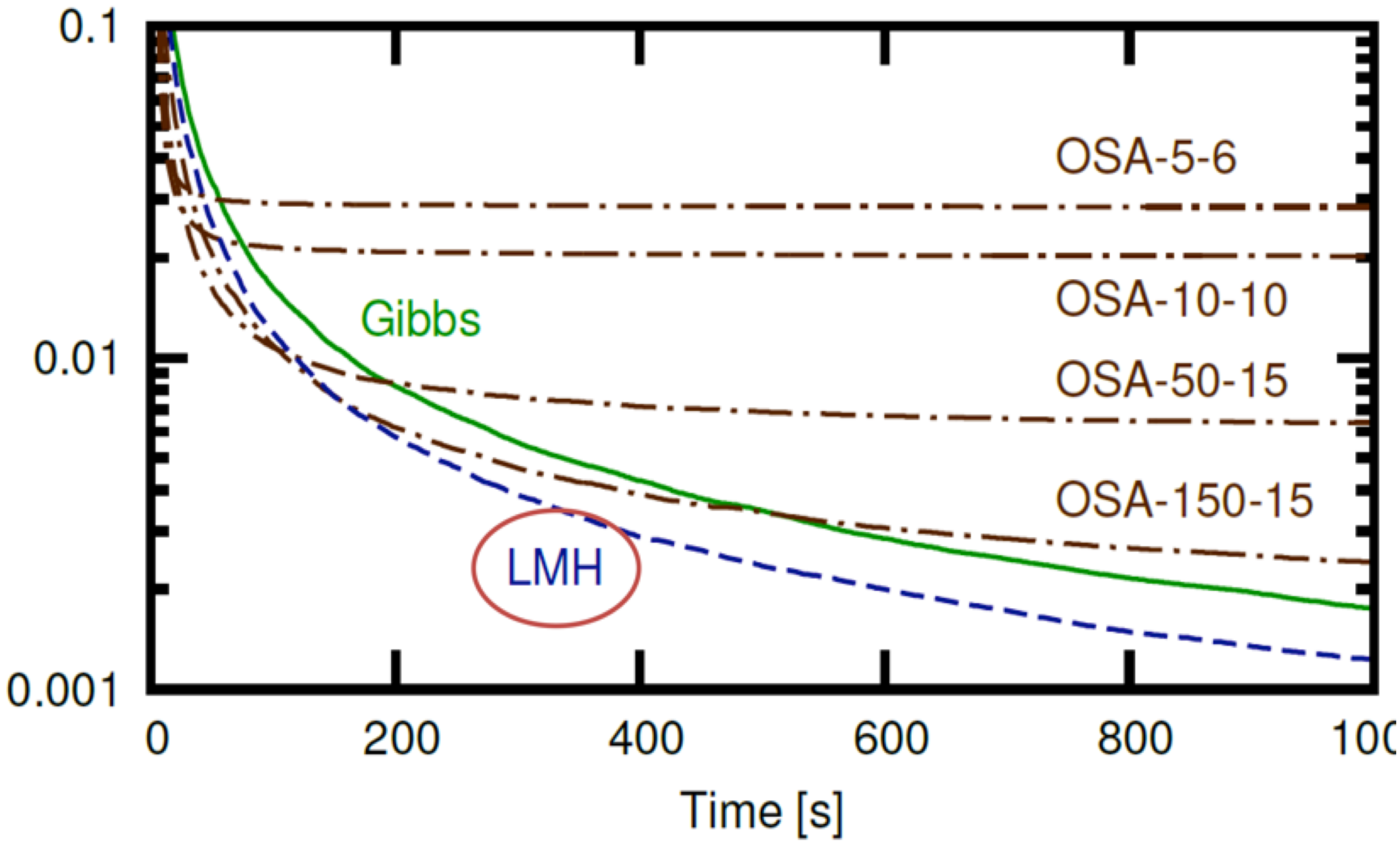
1. Select $\mathbf{y}$ uniformly at random from $\mathbf{x}^G$
2. Move from $\mathbf{x}$ to $\mathbf{y}$ with probability $\min\left(\frac{\Pr(\boldsymbol{y})}{\Pr(\boldsymbol{x})}, 1\right)$
3. Otherwise: stay in $\mathbf{x}$ (reject)
4. Repeat

and an ordinary (base) Markov chain B, with prob. α follow B and with (1-α) follow A

This can also account for evidence that may break symmetries, using e.g. approx. symmetries

[Niepert UAI 2012, Van den Broeck, Niepert AAAI 2015]

Lifted Metropolis-Hastings on WebKB



Take away

- Lifted inference exploits (fractional) symmetries
- Fractional symmetries can be computed in quasi-linear time
- Symmetries allow one to study lifted inference in an algebraic way, i.e., independent of the underlying algorithm
- Essentially, the whole family of approximate inference methods is liftable
- Lifted inference of interest to Optimization, ML, and AI in general (SVMs, RL, IRL, Deep Networks, …)