GIF German-Israeli Foundation for Scientific Research and Development

DFG

Martiin Mladenov

Leonard Kleinhans

Danny Heinrich

Kristian Kersting

http://www-ai.cs.uni-dortmund.de/weblab/static/RLP/html/

reloop

RELOOP: A Toolkit for Relational Convex Optimization

Luc De Raedt

Kristian Kersting

Sriraam Natarajan

David Poole

M&C Artificial Intelligence and Machine Learning

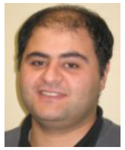
MORGAN & CLAYPOOL PUBLISHERS

Statistical Relational Artificial Intelligence

i


Martiin Mladenov


Pavel Tokmakov


Babak Ahmadi

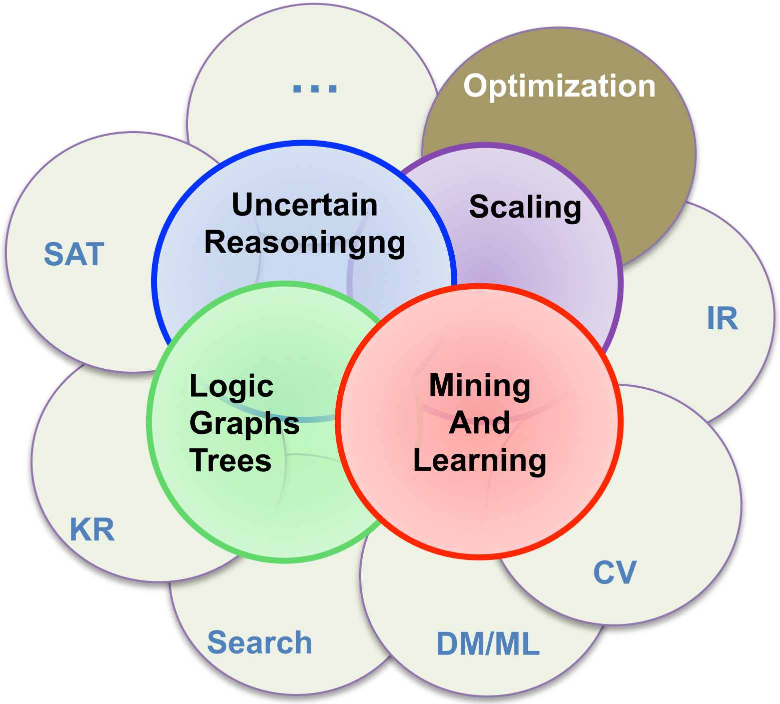

Sriraam Natarajan


Amir Globerson


Udi Apsel


Martin Grohe

and many more …



The Democratization of Optimization


Kristian Kersting

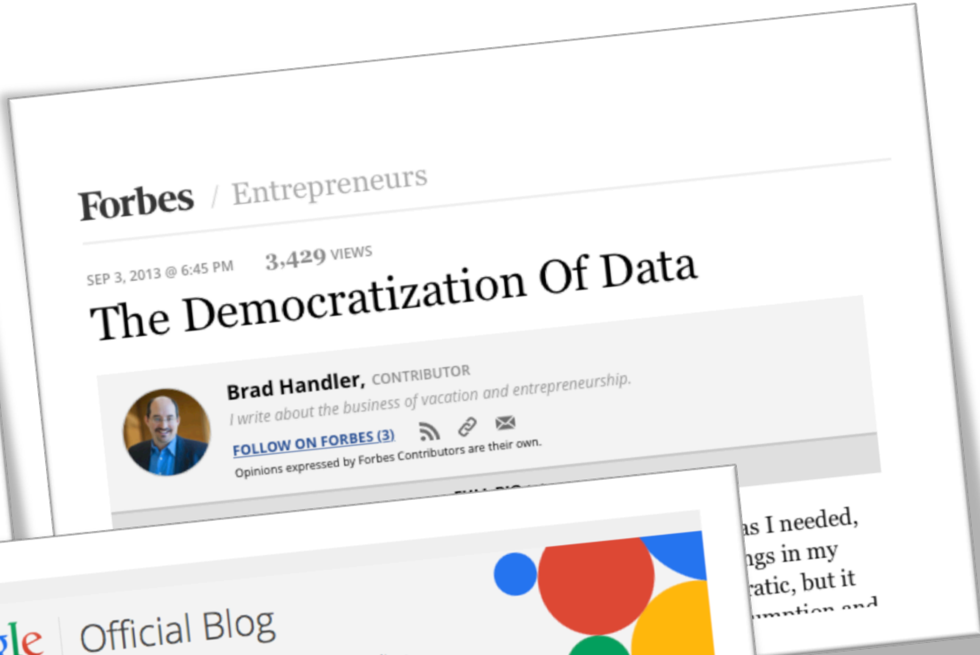
technische universität dortmund

GIF German-Israeli Foundation for Scientific Research and Development

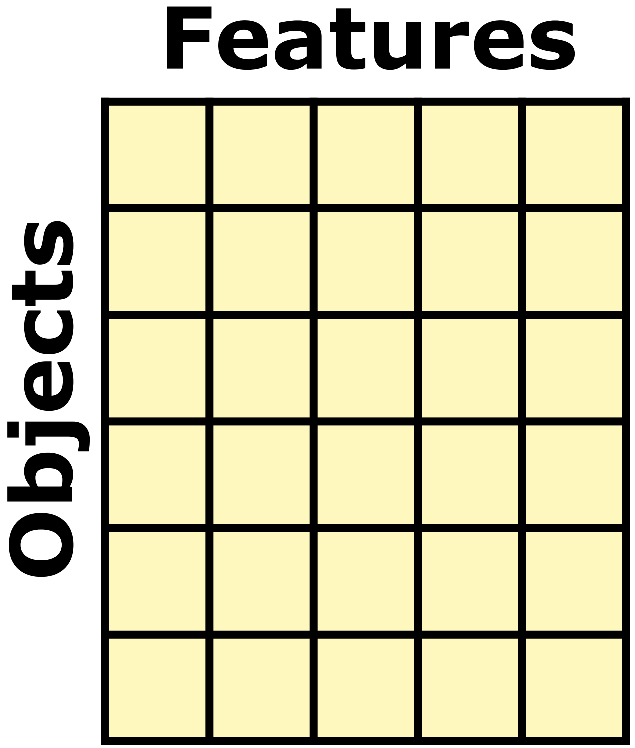
DFG

SEVENTH FRAMEWORK PROGRAMME

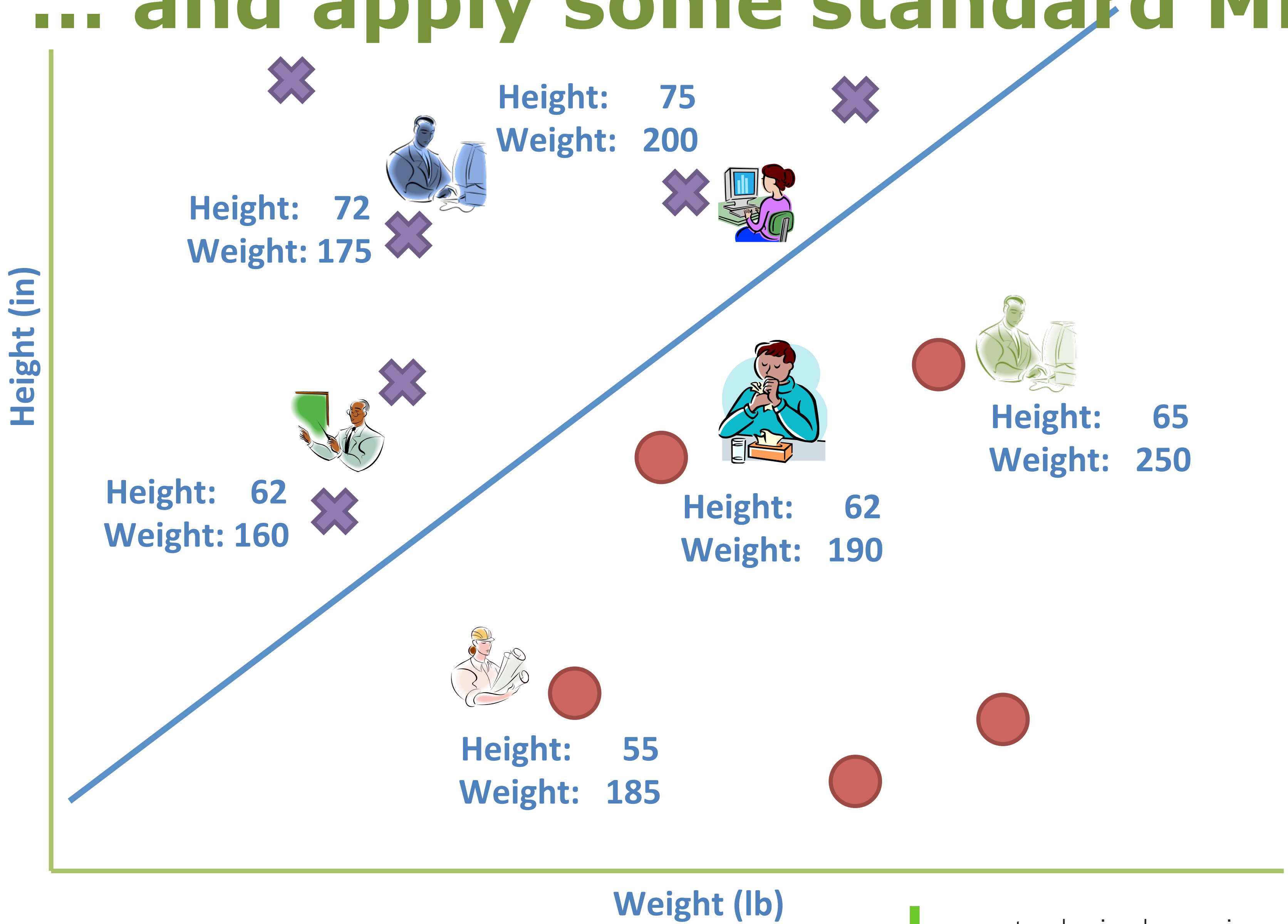
Democratization of Data



Take your spreadsheet …

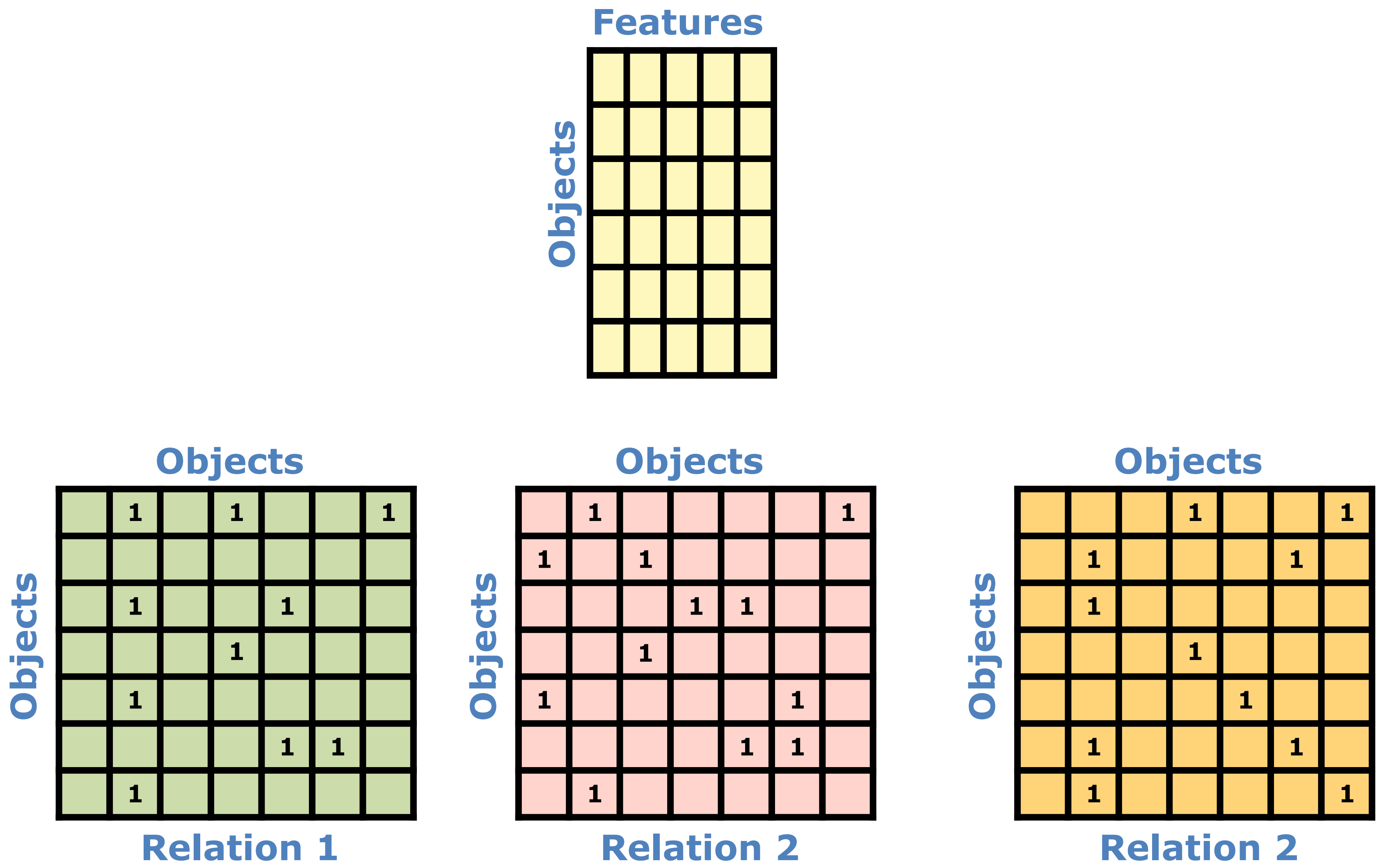


… and apply some standard ML



IS IT REALLY THAT SIMPLE?

NO, e.g., today's data is relational



Nat Rev Genet. 2012 May 2;13(6):395-405

Heart diseases and strokes – cardiovascular disease – are expensive for the world

According to the World Heart Federation, cardiovascular disease cost the European Union EURO169 billion in 2003 and the USA about EURO310.23 billion in direct and indirect annual costs. By comparison, the estimated cost of all cancers is EURO146.19 billion and HIV infections, EURO22.24 billion

Electronic Health Records A New Opportunity for AI to Save our Lifes

[Natarajan, Kersting, et al. IAAI 2013, Springer Briefs in CS 2015, AIME 2015]

EHRs are dirty and interconnected

Patient Table

PatientID	Gender	Birthdate
P1	M	3/22/63

Visit Table

PatientID	Date	Physician	Symptoms	Diagnosis
P1	1/1/01	Smith	palpitations	hypoglycemic
P1	2/1/03	Jones	fever, aches	influenza

Lab Tests

PatientID	Date	Lab Test	Result
P1	1/1/01	blood glucose	42
P1	1/9/01	blood glucose	??

SNP Table

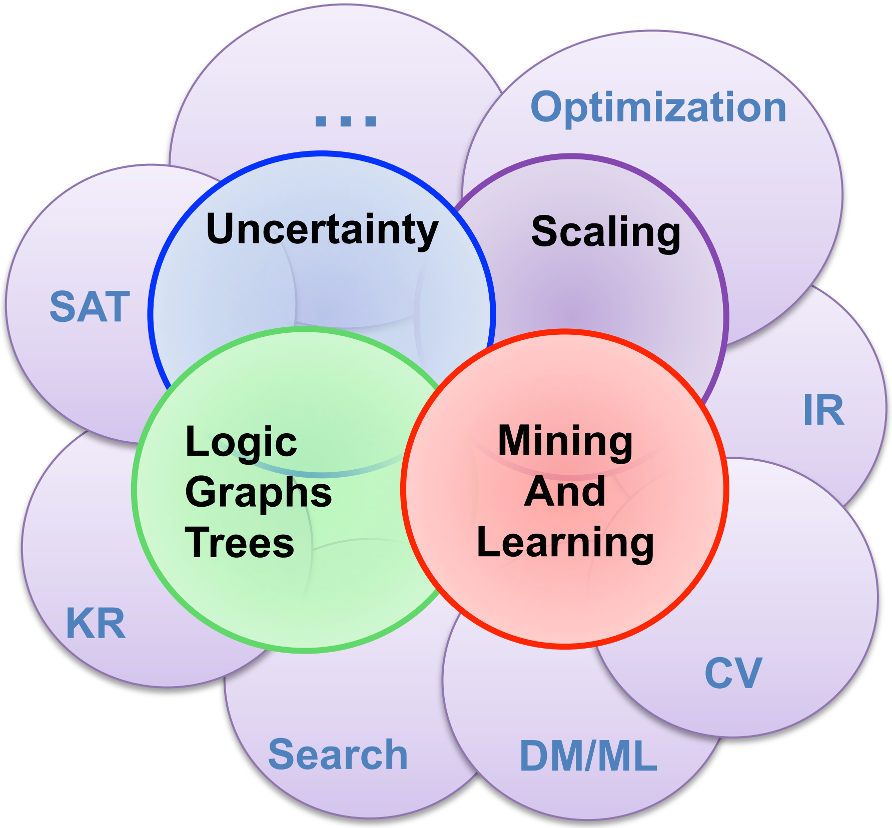
PatientID	SNP1	SNP2	…	SNP500K
P1	AA	AB		BB
P2	AB	BB		AA

Prescriptions

PatientID	Date Prescribed	Date Filled	Physician	Medication	Dose	Duration
P1	5/17/98	5/18/98	Jones	prilosec	10mg	3 months

Statistical Relational AI ...

… **the study and design of intelligent agents that act in noisy worlds composed of objects and relations among the objects**



Thanks to you - the Iatlian AI community - for your great contributions!

technische universität dortmund

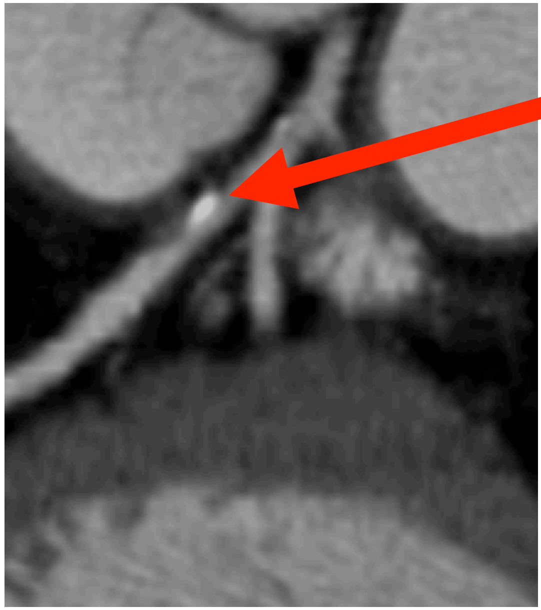
[Kersting, Driessens ICML 2008; Karwath, Kersting, Landwehr ICDM 2008; Natarajan, Joshi, Tadepelli, Kersting, Shavlik. IJCAI 2011; Khot, Natarajan, Kersting, Shavlik ICDM 2013, MLJ 2012, Springer Brief 2015, MLJ 2015]

Boosted Statistical Relational Learning

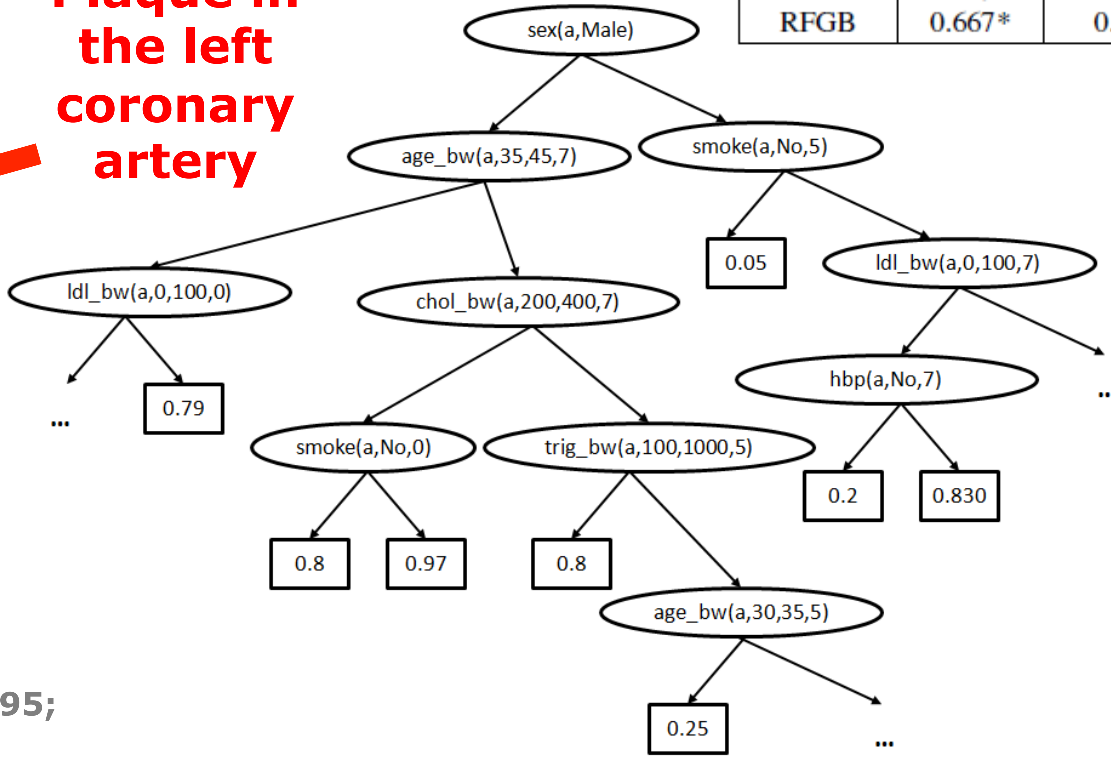
Atherosclerosis is the cause of the majority of Acute Myocardial Infarctions (heart attacks)

Algorithm	Accuracy	AUC-ROC
J48	0.667	0.607
SVM	0.667	0.5
AdaBoost	0.667	0.608
Bagging	0.677	0.613
NB	0.75	0.653
RPT	0.669*	0.778
RFGB	0.667*	0.819

Plaque in the left coronary artery



[Circulation; 92(8), 2157-62, 1995; JACC; 43, 842-7, 2004]



Algo	Likelihood	AUC-ROC	AUC-PR	Time
Boosting	0.810	0.961	0.930	**9s**
MLN	0.730	0.535	0.621	**93 hrs**

Take-Away Messages

1. **Graphical models allow to deal with uncertainty and to make predictions**
2. **Graphs/Matrices are not enough, we need logic/high-level languages**

STILL NOT CONVINCED?

Guy van den Broeck’s not so simple AI example

What is the probability that the first card of a shuffled deck is an Ace?

Easy for humans but not so easy for graphical models

Exact inference builds a table of $\geq 13^{52}$ rows!

Message passing passes $\geq 13^{52}$ messages!

Graphical model is fully connected, no independencies,high tree-width

Low tree-width is not the final answer

Fast modelling

Fast inference

We need relational models

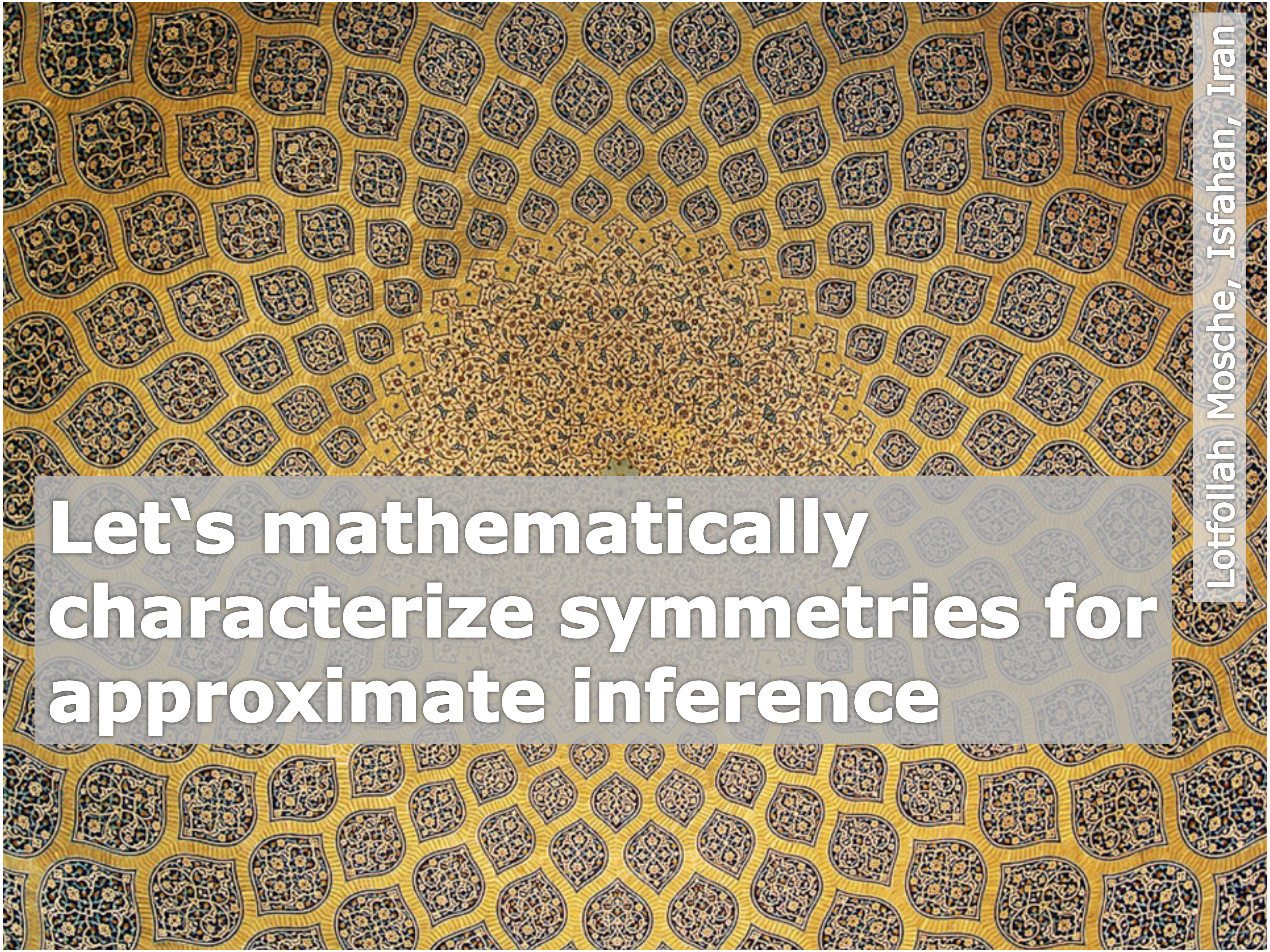
w1 : $\forall$p,x,y, Card(p,x) $\wedge$ Card(p,y) $\Rightarrow$ x = y

w2 : $\forall$c,x,y, Card(x,c) $\wedge$ Card(y,c) $\Rightarrow$ x = y

and symmetry-reduction

Take-away Messages

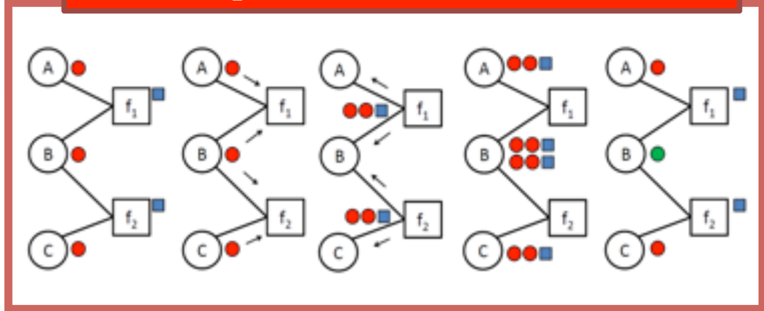
1. Graphical models allow to deal with uncertainty
2. Graphs/Matrices are not enough, we need logic /high-level languages
3. Tree-Width is not the end of the story

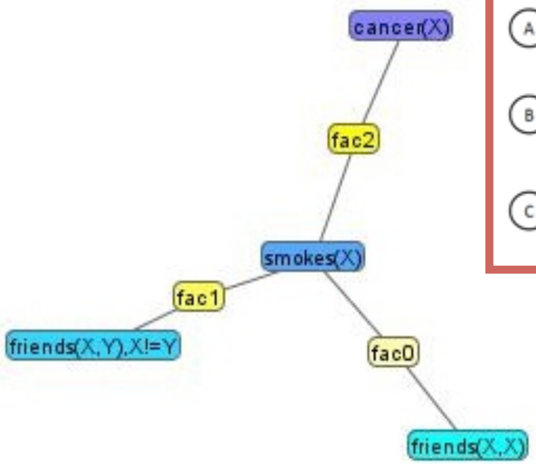

Let`s mathematically characterize symmetries for approximate inference
Lotfollah Mosche, Isfahan, Iran

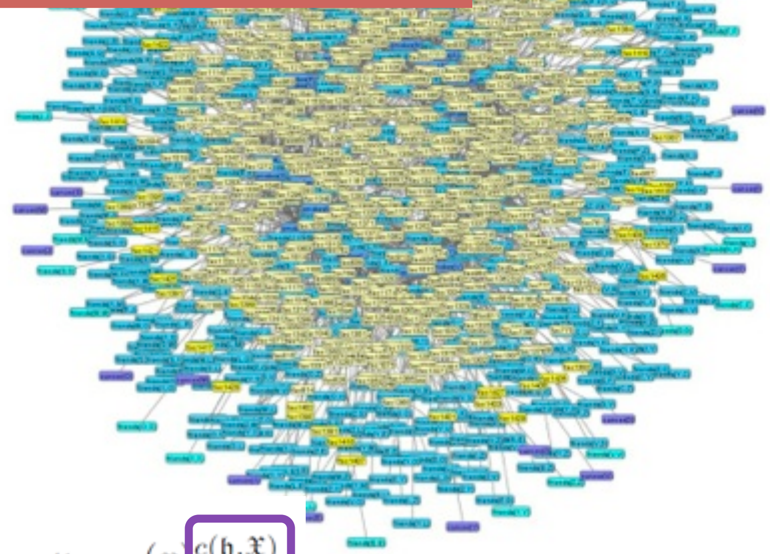
[Singla, Domingos AAAI 2008; Kersting, Ahmadi, Natarajan UAI 2009; Ahmadi, Kersting, Mladenov, Natarajan MLJ 2013]

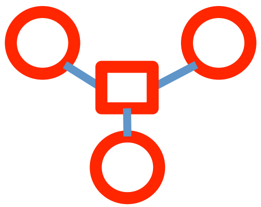
Lifted Loopy Belief Propagation = Exploit computational symmetries

Compress the model







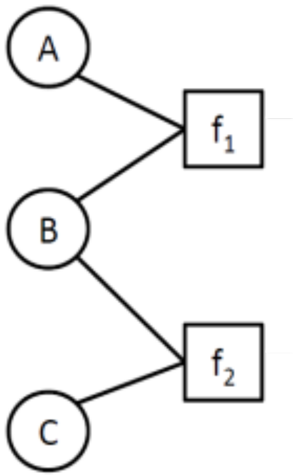


$$b_i(x_i) = \prod_{\mathfrak{f}\in \mathrm{nb}(\mathfrak{X}_i)} \mu_{\mathfrak{f}\to\mathfrak{X}_i}(x_i)^{\boxed{c(\mathfrak{f},\mathfrak{X})}}$$

$$\mu_{\mathfrak{X}\to\mathfrak{f}}(x) = \mu_{\mathfrak{f}\to\mathfrak{X}}(x)^{\boxed{c(\mathfrak{f},\mathfrak{X})-1}} \cdot \prod_{\mathfrak{h}\in \mathrm{nb}(\mathfrak{X})\setminus\{\mathfrak{f}\}} \mu_{\mathfrak{h}\to\mathfrak{X}}(x)^{\boxed{c(\mathfrak{h},\mathfrak{X})}}$$

Run a modified Loopy Belief Propagation

Compression: Coloring the graph

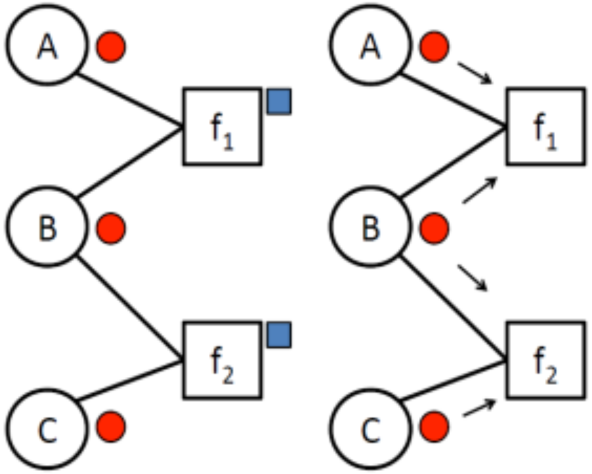


- **Color nodes according to the evidence you have**
  - No evidence, say **red**
  - State „one", say **brown**
  - State „two", say **orange**
  - ...
- **Color factors distinctively according to their equivalence classes.** For instance, assuming f_1 and f_2 to be identical and B appears at the second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

technische universität dortmund

[Singla, Domingos AAAI 2008; Kersting, Ahmadi, Natarajan UAI 2009; Ahmadi, Kersting, Mladenov, Natarajan MLJ 2013]

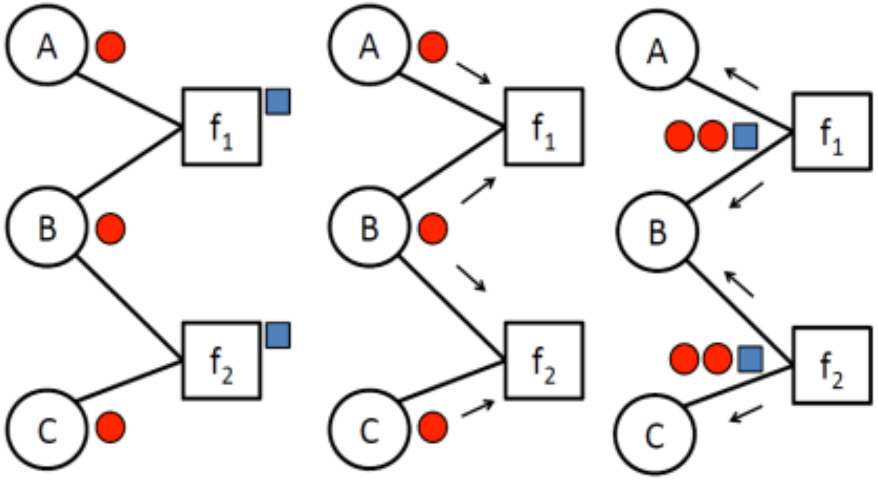
Compression: Pass the colors around



1. Each factor collects the colors of its neighboring nodes

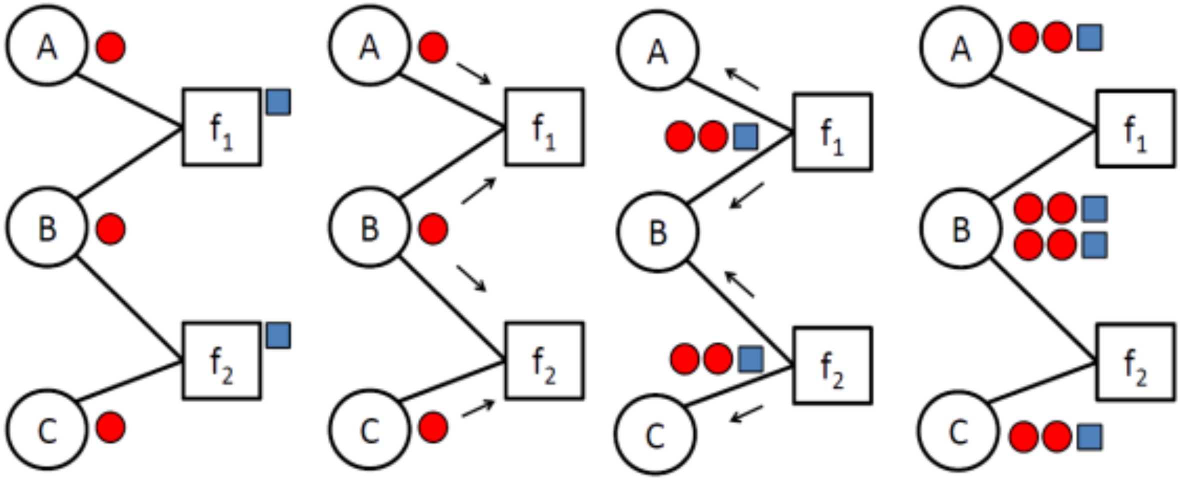
[Singla, Domingos AAAI 2008; Kersting, Ahmadi, Natarajan UAI 2009; Ahmadi, Kersting, Mladenov, Natarajan MLJ 2013]

Compression: Pass the colors around



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

Compression: Pass the colors around

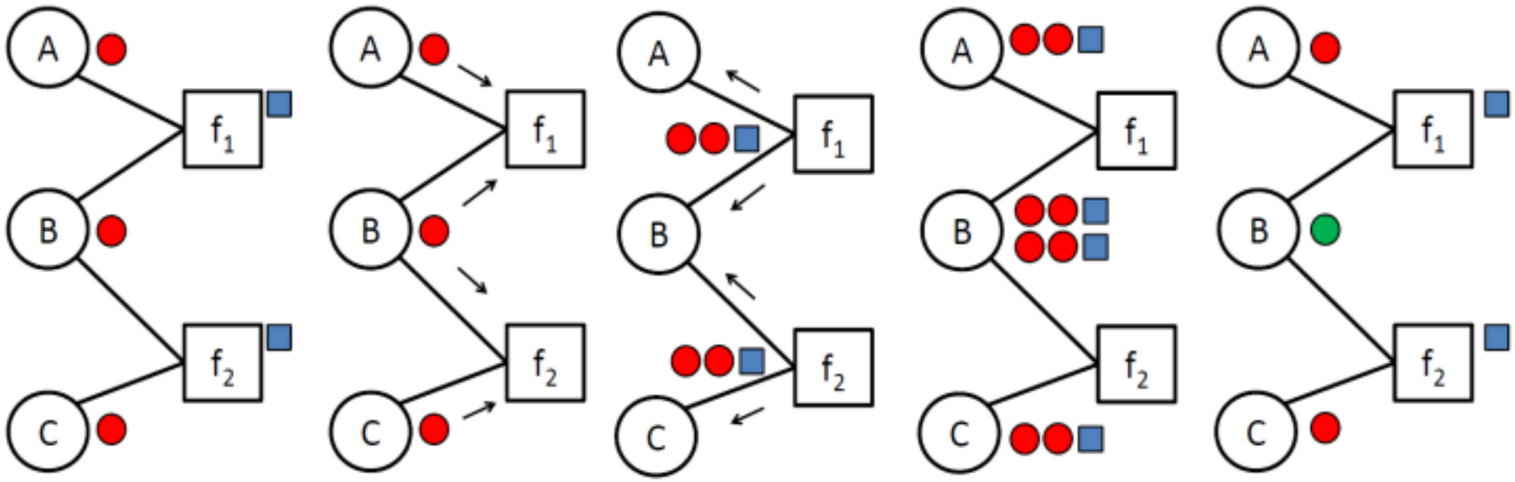


1. Each factor collects the colors of its neighboring nodes
2. Each factor „signs" ist color signature with its own color
3. Each node collects the signatures of its neighboring factors

technische universität dortmund

[Singla, Domingos AAAI 2008; Kersting, Ahmadi, Natarajan UAI 2009; Ahmadi, Kersting, Mladenov, Natarajan MLJ 2013]

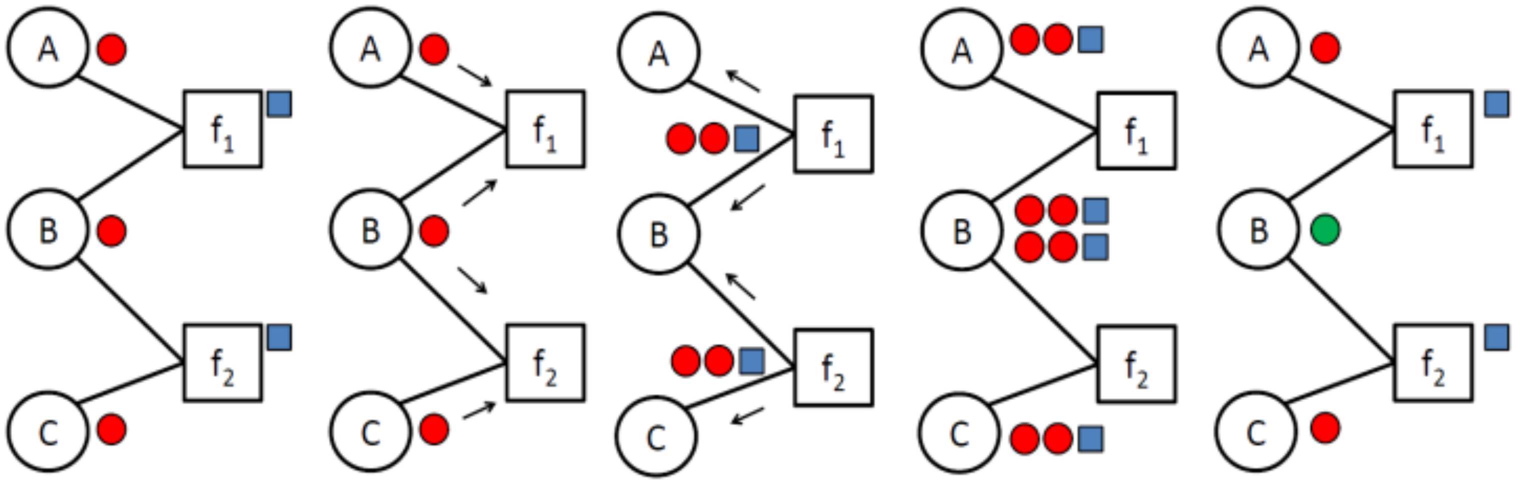
Compression: Pass the colors around



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

[Singla, Domingos AAAI 2008; Kersting, Ahmadi, Natarajan UAI 2009; Ahmadi, Kersting, Mladenov, Natarajan MLJ 2013]

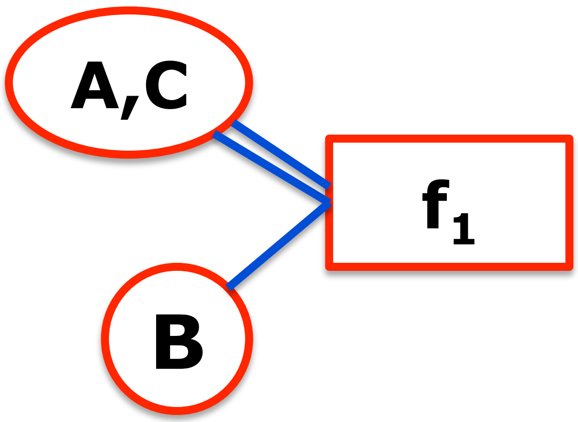
Compression: Pass the colors around

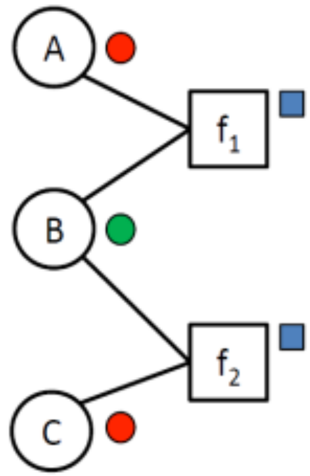


1. Each factor collects the colors of its neighboring nodes
2. Each factor „signs" ist color signature with its own color
3. Each node collects the signatures of its neighboring factors
4. Nodes are recolored according to the collected signatures
5. If no new color is created stop, otherwise go back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

[Singla, Domingos AAAI 2008; Kersting, Ahmadi, Natarajan UAI 2009; Ahmadi, Kersting, Mladenov, Natarajan MLJ 2013]

Compression:
… and compute the quotient factor graph



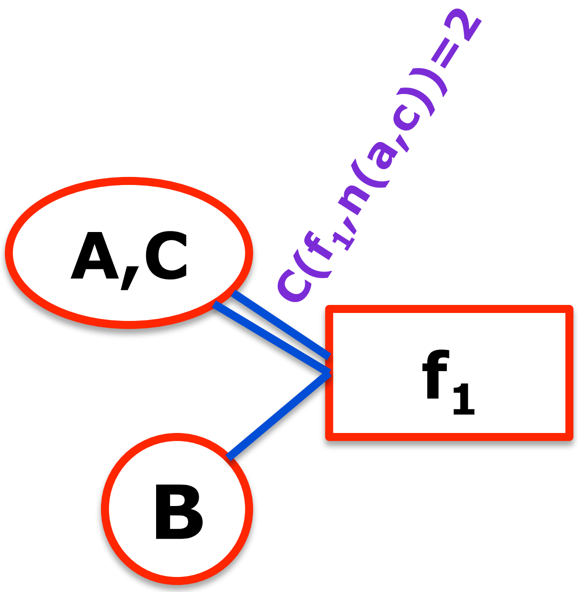


Essentially we just compute the so-called quotient factor graph

technische universität dortmund

[Singla, Domingos AAAI 2008; Kersting, Ahmadi, Natarajan UAI 2009; Ahmadi, Kersting, Mladenov, Natarajan MLJ 2013]

Finally, run a modified Loopy Belief Propagation



$$b_i(x_i) = \prod_{\mathfrak{f} \in \mathrm{nb}(\mathfrak{X}_i)} \mu_{\mathfrak{f}\to\mathfrak{X}_i}(x_i)^{\boxed{c(\mathfrak{f},\mathfrak{X})}}$$

$$\mu_{\mathfrak{X}\to\mathfrak{f}}(x) = \mu_{\mathfrak{f}\to\mathfrak{X}}(x)^{\boxed{c(\mathfrak{f},\mathfrak{X})-1}} \cdot \prod_{\mathfrak{h}\in \mathrm{nb}(\mathfrak{X})\setminus\{\mathfrak{f}\}} \mu_{\mathfrak{h}\to\mathfrak{X}}(x)^{\boxed{c(\mathfrak{h},\mathfrak{X})}}$$

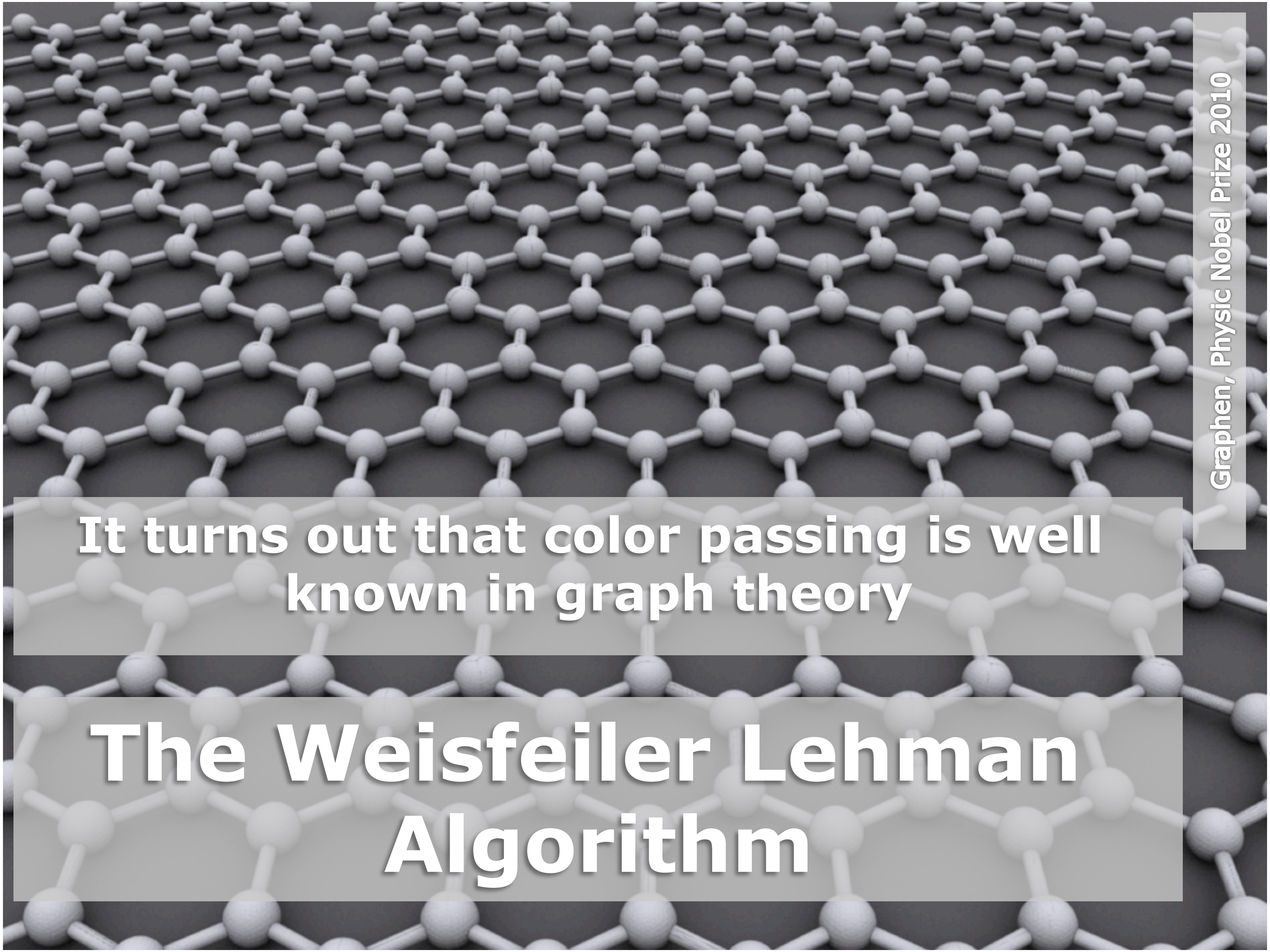
- Nodes are now groups of random variables
- The **counts** ensure that we send the same number of message as standard loopy belief propagation

Lifted Probabilistic Inference

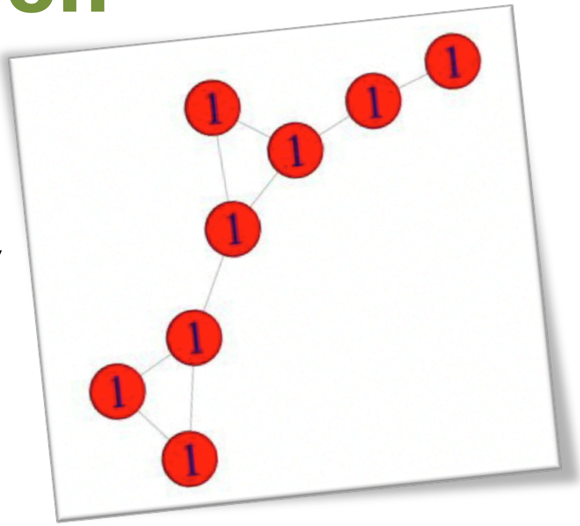
Compress the model

Run a modified inference method

might also be interwined

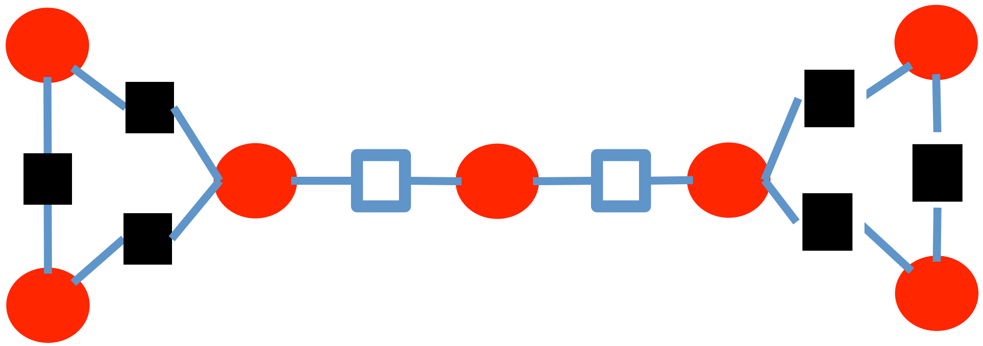

Graphen, Physic Nobel Prize 2010
It turns out that color passing is well known in graph theory
The Weisfeiler Lehman Algorithm

Weisfeiler-Lehman (WL) Algorithmus aka "naive vertex classification"

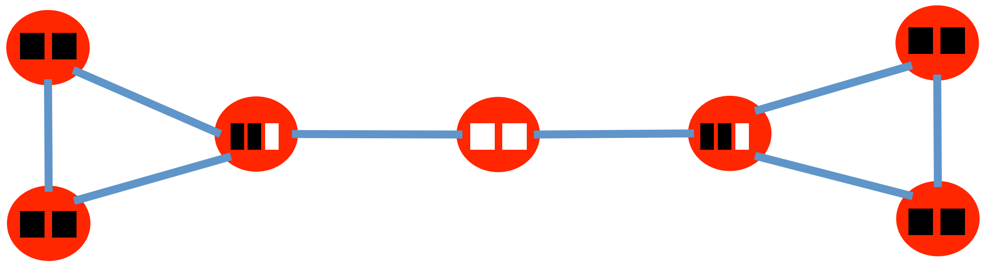


- Basic subroutine for GI testing
- Computes LP-relaxations of GA-ILP, aka. **fractional automorphisms**
- **Quasi-linear** running time O((n+m)log(n)) when using asynchronous updates [Berkholz, Bonsma, Grohe ESA 2013]
- **Part of graph tool SAUCY** [See e.g. Darga, Sakallah, Markov DAC 2008]
- Can be extended to weighted graphs/real-valued matrices [Grohe, Kersting, Mladenov, Selman ESA 2014]
- Actually a Frank-Wolfe optimizer and can be viewed as recursive spectral clustering [Kersting, Mladenov, Garnett, Grohe AAAI 2014]

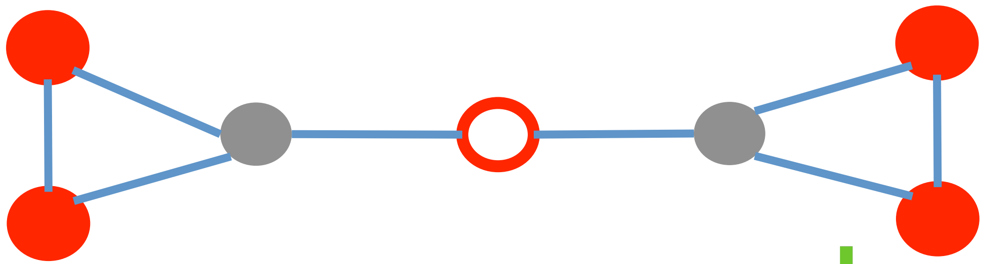
From Factor Graphs to Graphs



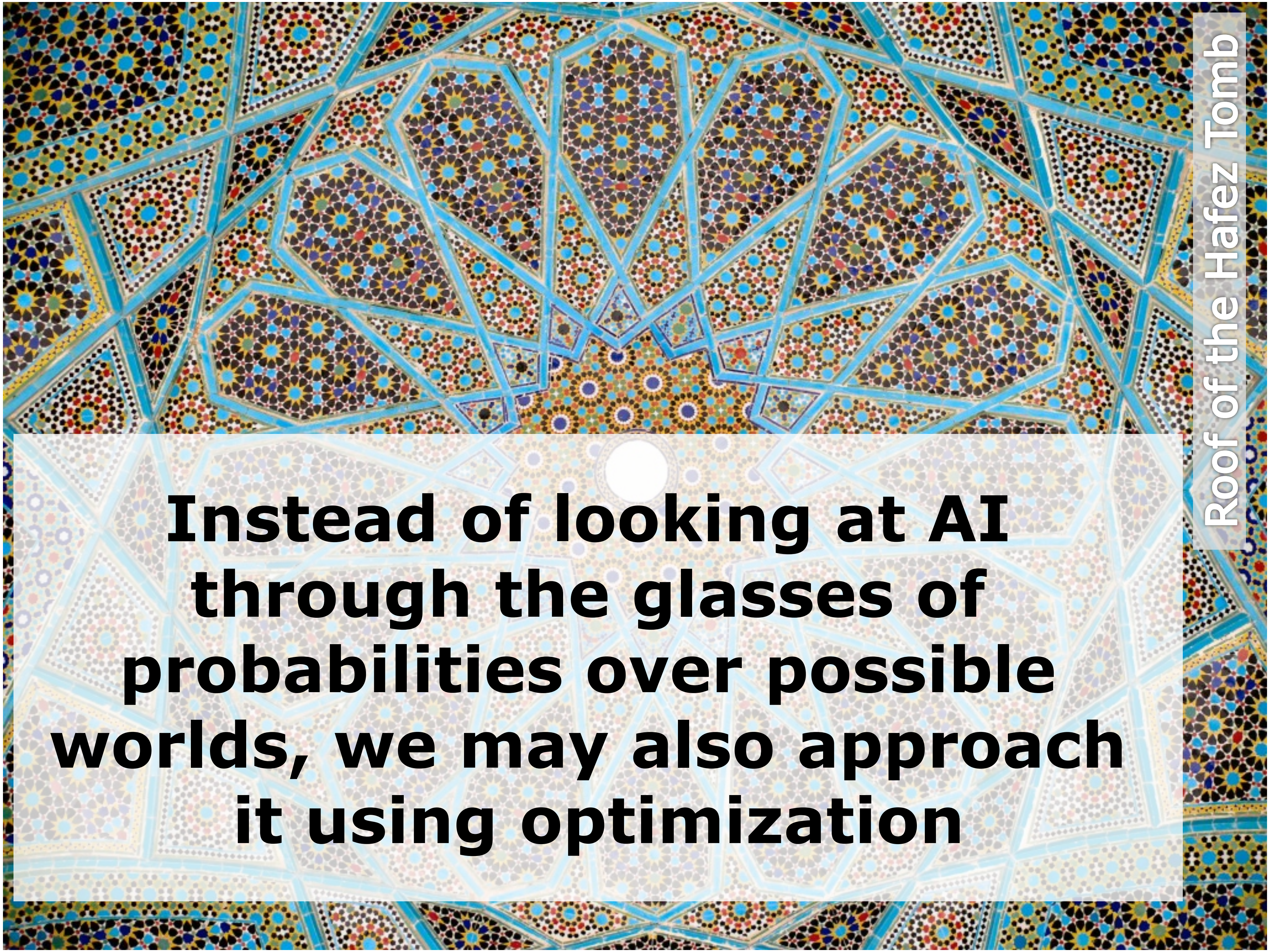
Encode the factor colors into the node colors



Then run Weisfeiler Lehman / Color Passing just on the graph



technische universität dortmund



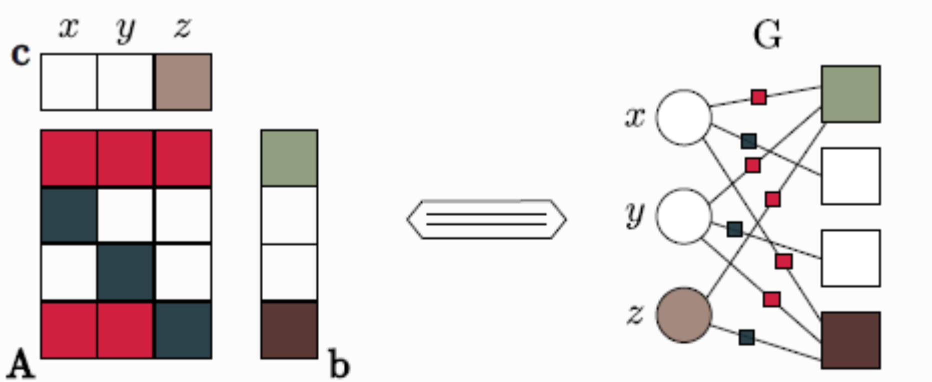
Instead of looking at AI through the glasses of probabilities over possible worlds, we may also approach it using optimization

[Mladenov, Ahmadi, Kersting AISTATS 2012, Grohe, Kersting, Mladenov, Selman ESA 2014, Kersting Mladenov, Tokmatov AIJ 2015]

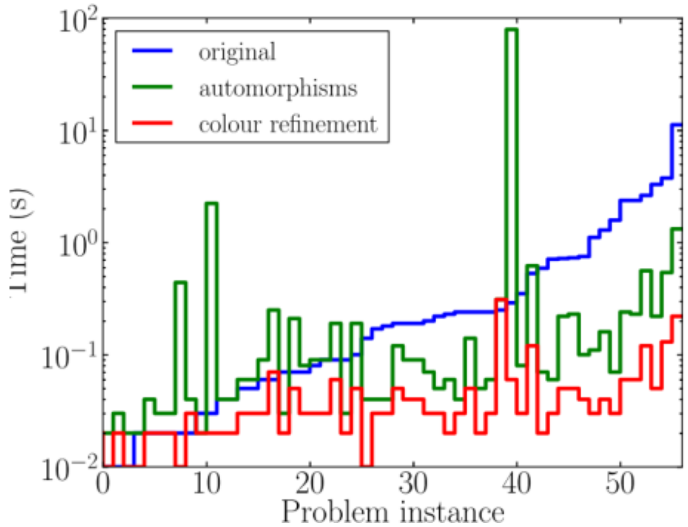
Compressing Linear Programs

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

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



(1) Reduce the LP by running WL on the LP-Graph
(2) Run any solver on the (hopefully) smaller LP

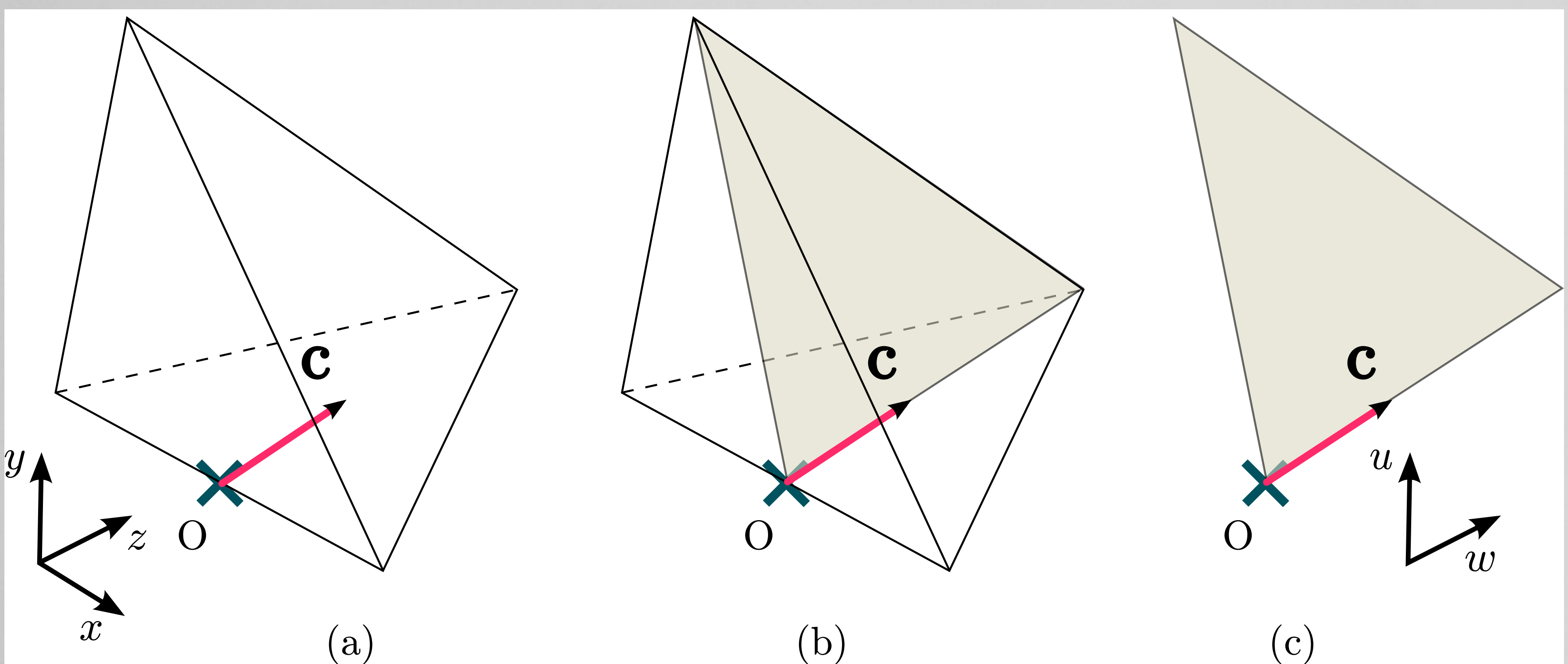


quasi-linear overhead that may result in exponential speed up

technische universität dortmund

As also noted by Stephen Boyd

DENSE VS. SPARSE IS NOT ENOUGH, SOLVERS NEED TO BE AWARE OF SYMMETRIES



Feasible region of LP and the objective vectors

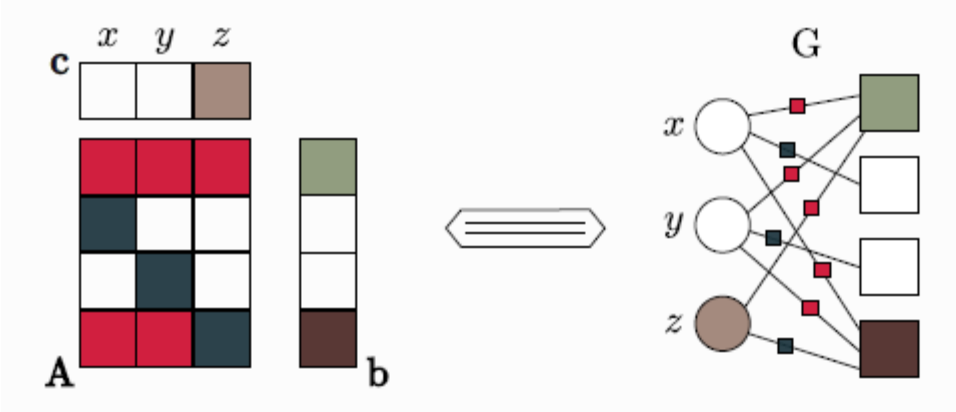
Span of the fractional automorpishm of the LP

Projections of the feasible region onto the span of the fractional automorphism

Why does this work?

[Mladenov, Ahmadi, Kersting AISTATS 2012, Grohe, Kersting, Mladenov, Selman ESA 2014, Kersting Mladenov, Tokmatov AIJ 2015]

Compute Equitable Partition (EO) 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.

technische universität dortmund

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

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

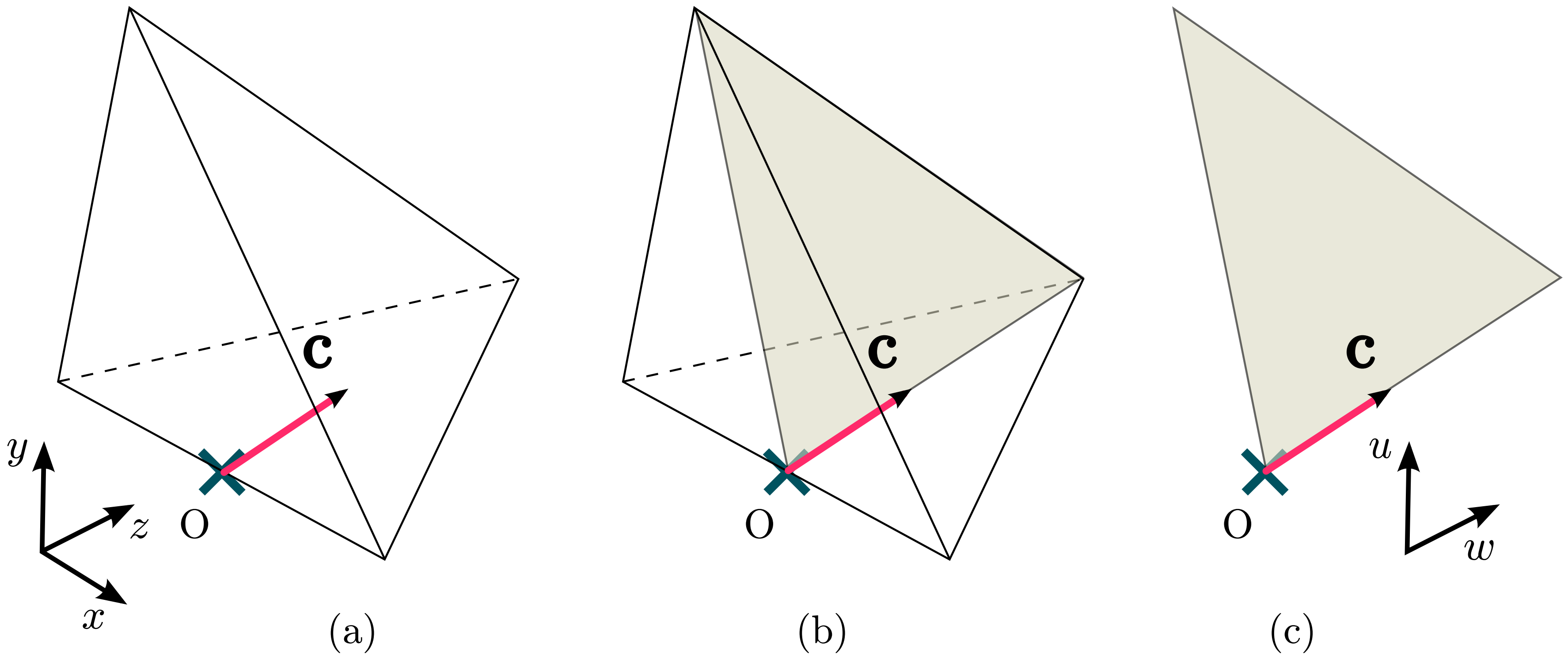
$$\mathbf{B}_{iP} = \begin{cases} \frac{1}{\sqrt{|P|}} & \text{if vertex } i \text{ belongs to part } P, \\ 0 & \text{otherwise.} \end{cases}$$

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 constraint that becomes identical

Dimensionality Reduction of LPs

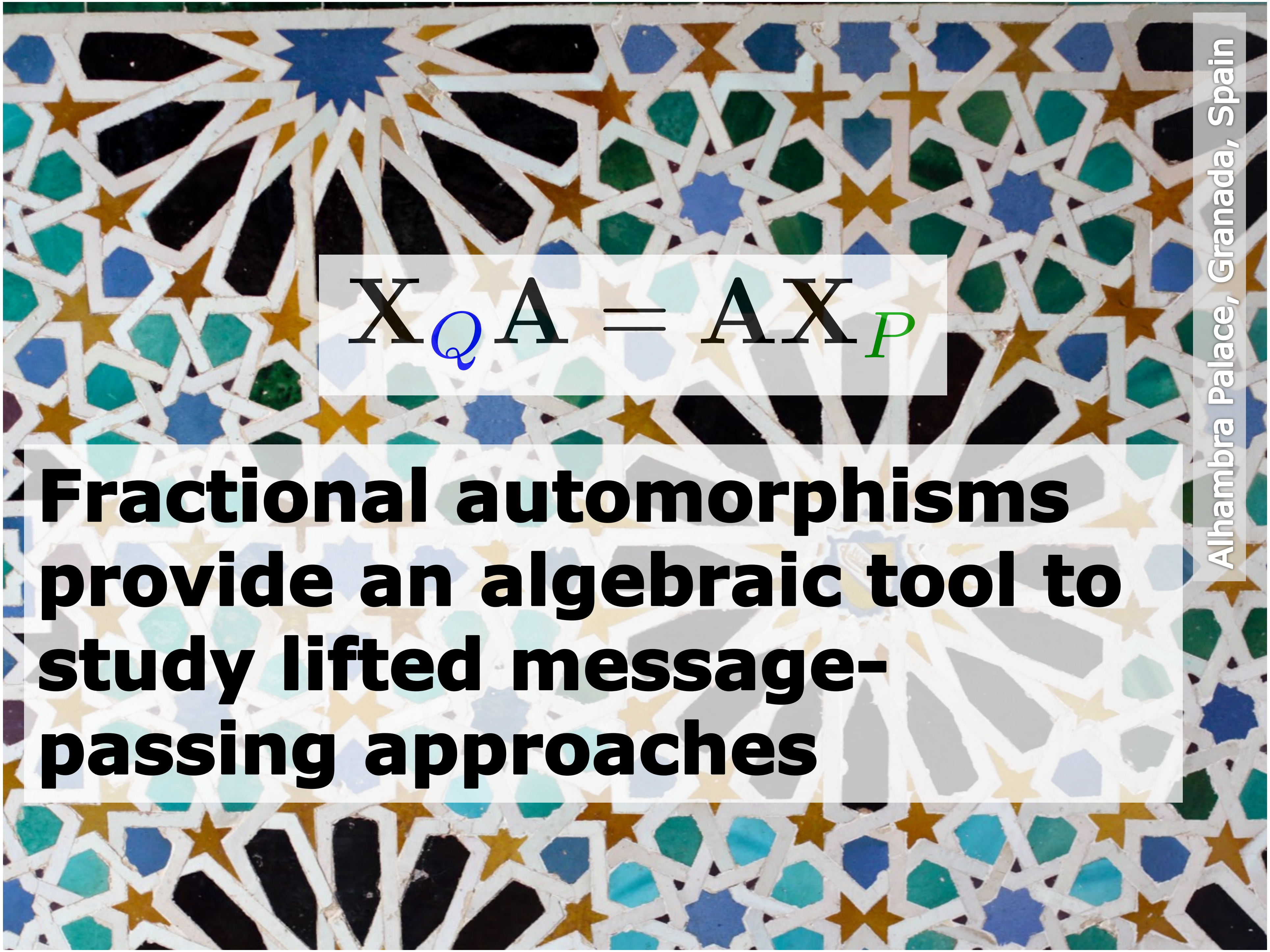


Feasible region of LP and the objective vectors

Span of the fractional automorphism of the LP

Projections of the feasible region onto the span of the fractional automorphism

technische universität dortmund

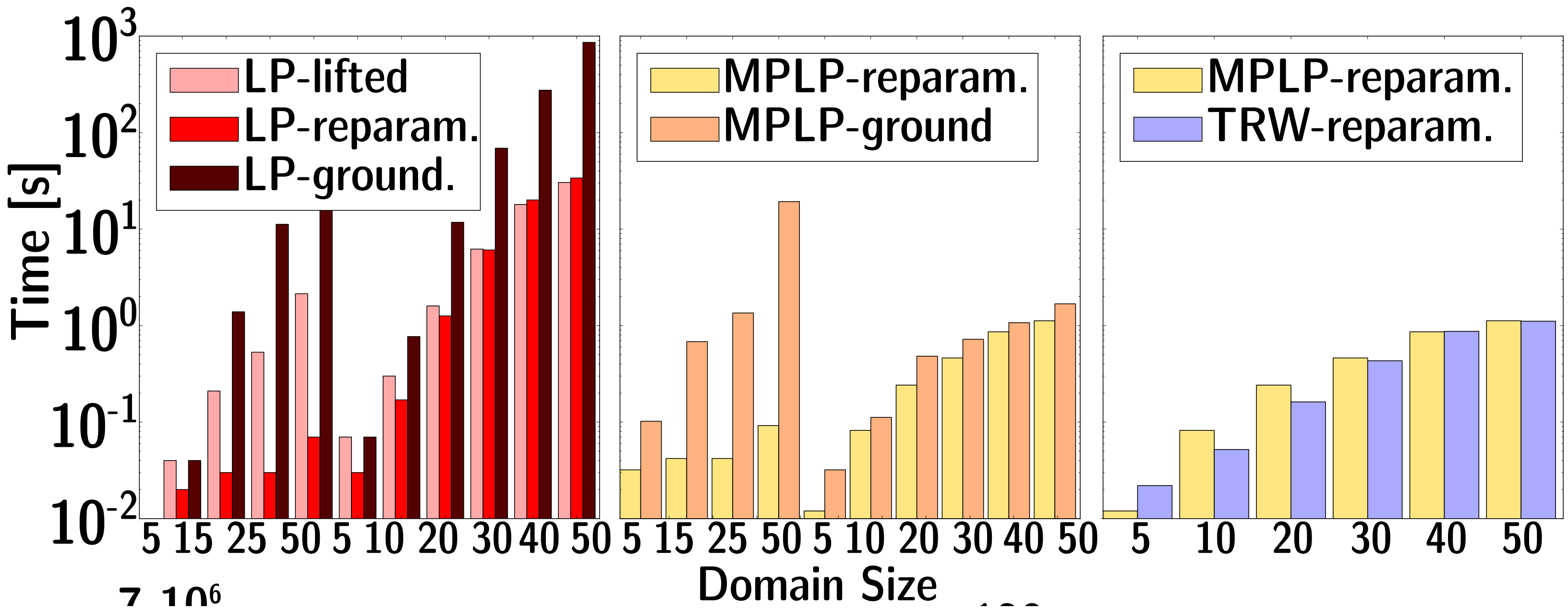


Alhambra Palace, Granada, Spain

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

Fractional automorphisms provide an algebraic tool to study lifted message-passing approaches

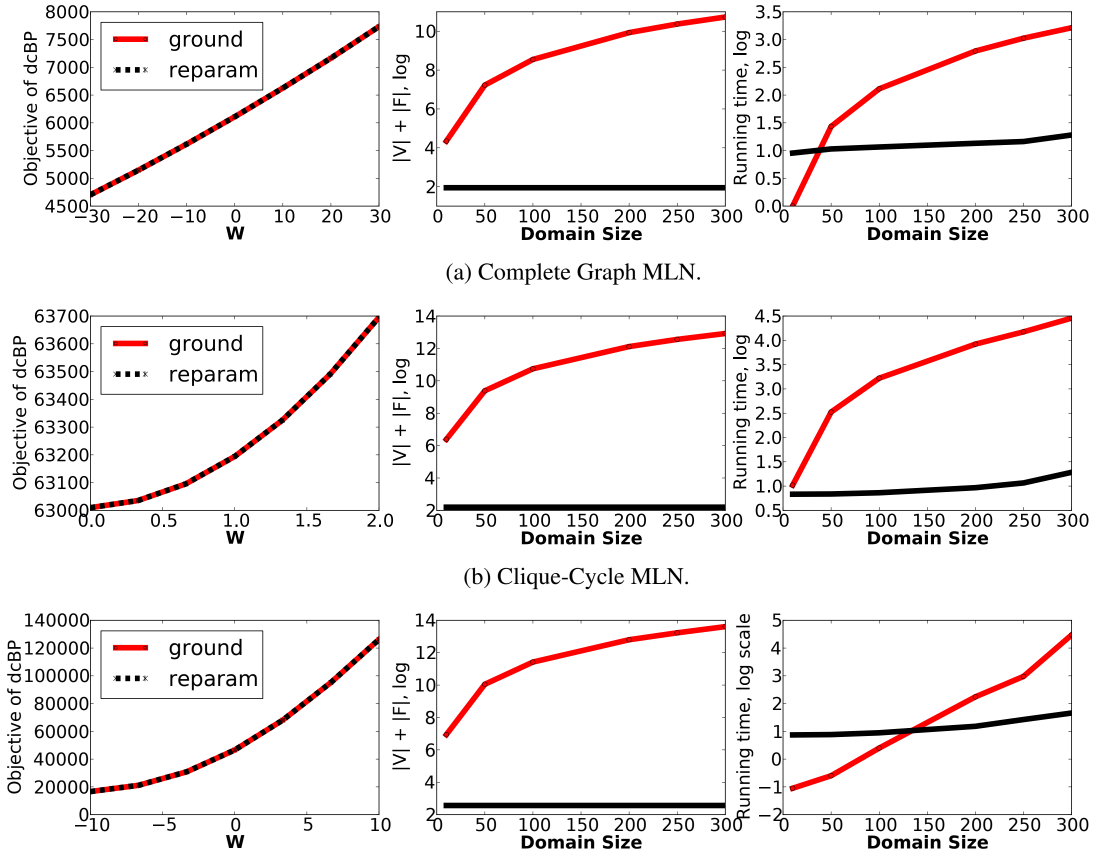
Any MAP-LP message-passing approach is liftable



technische universität dortmund

[Mladenov, Kersting UAI 2015]

Any concave free energy is liftable



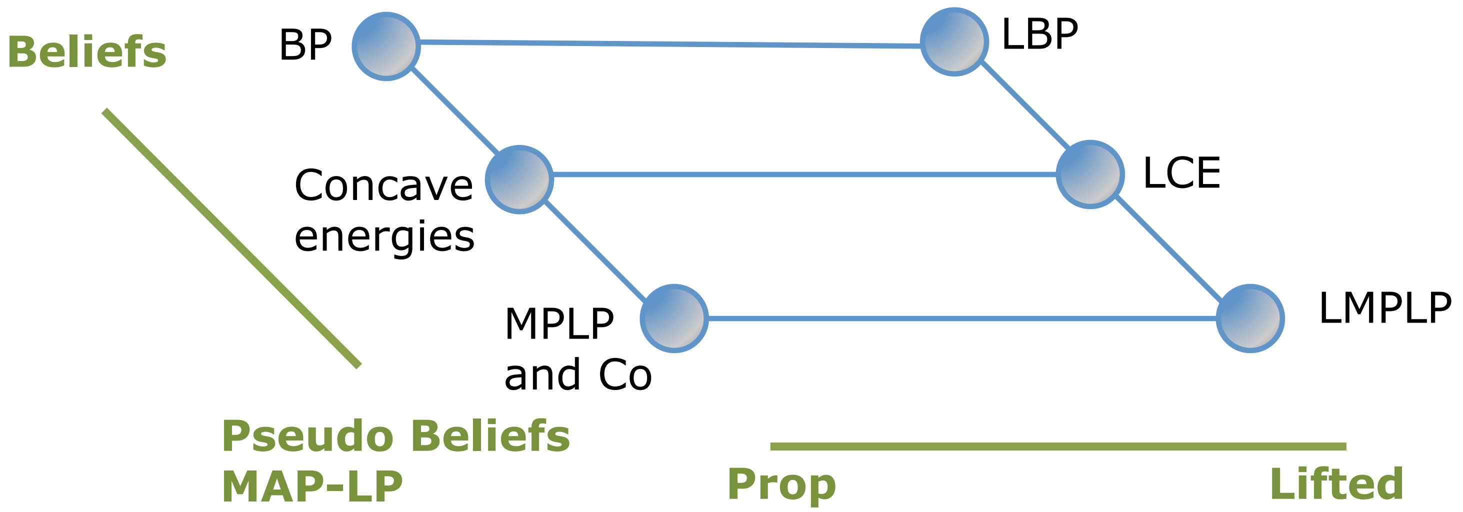
(a) Complete Graph MLN.

(b) Clique-Cycle MLN.

(c) Friends-smokers MLN.

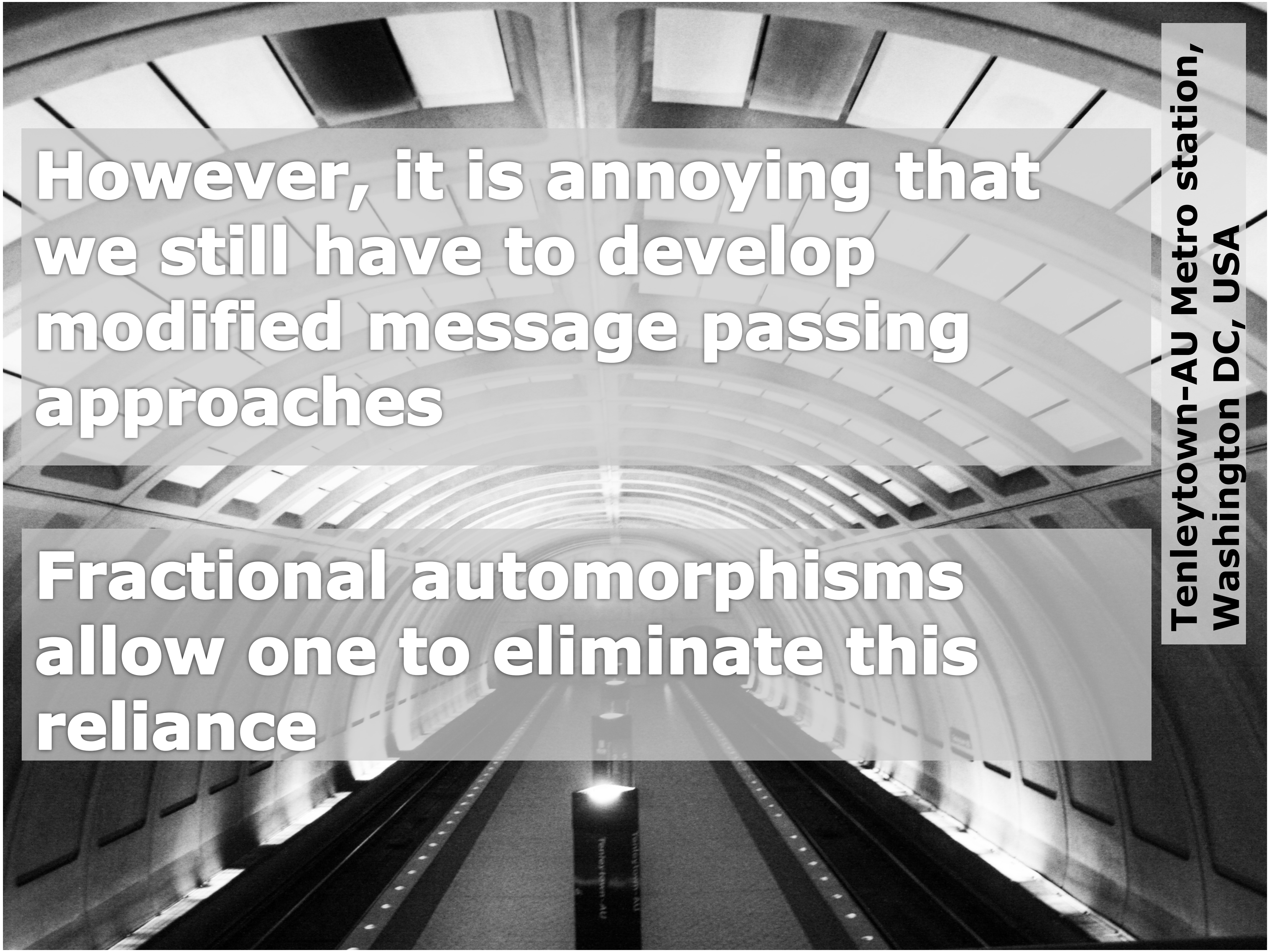
Actually, this is the first distributed lifted message-passing approach

All MP Inference Approaches are Liftable



However, it is annoying that we still have to develop modified message passing approaches

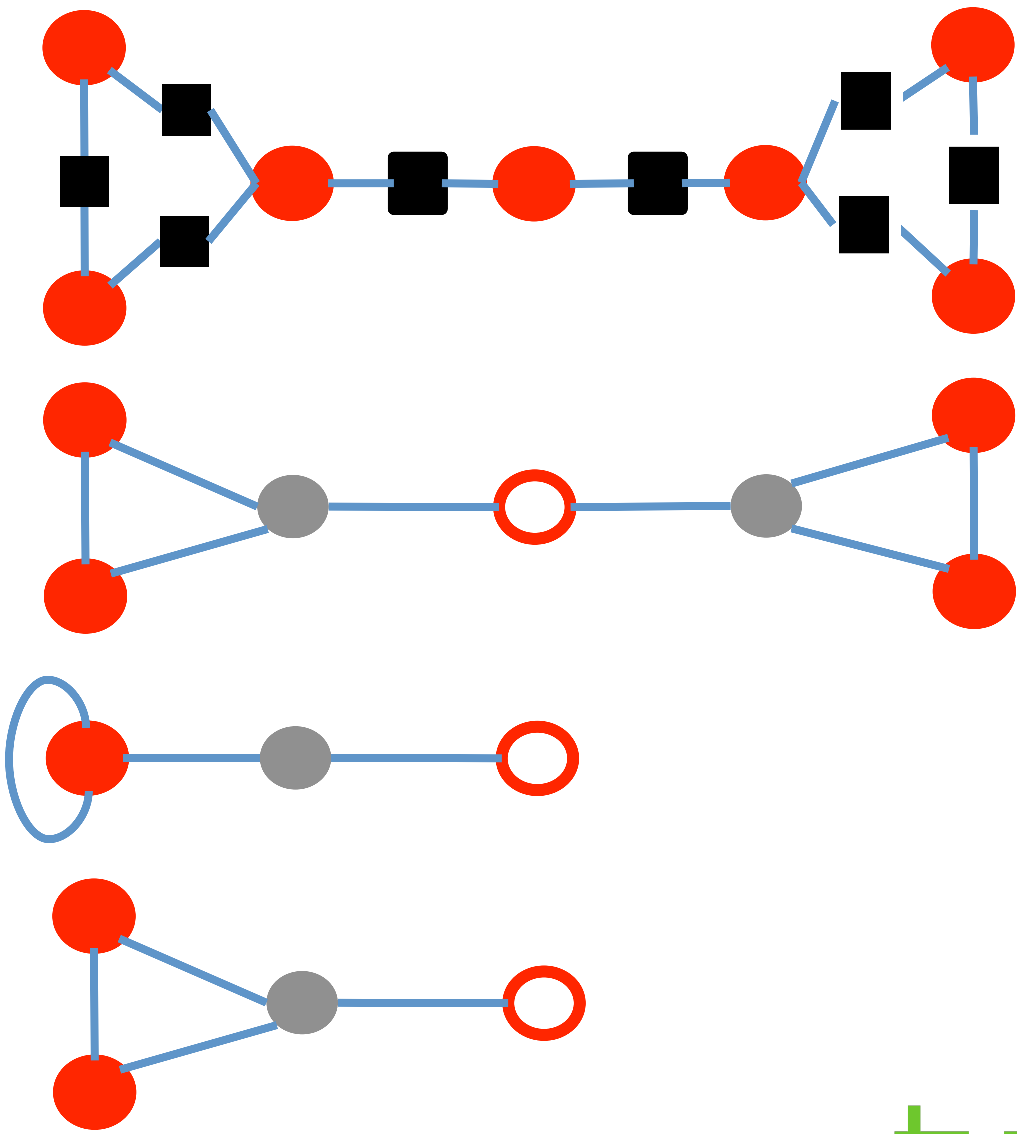
Fractional automorphisms allow one to eliminate this reliance



Tenleytown-AU Metro station, Washington DC, USA

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

Key Idea: Refine self-loops

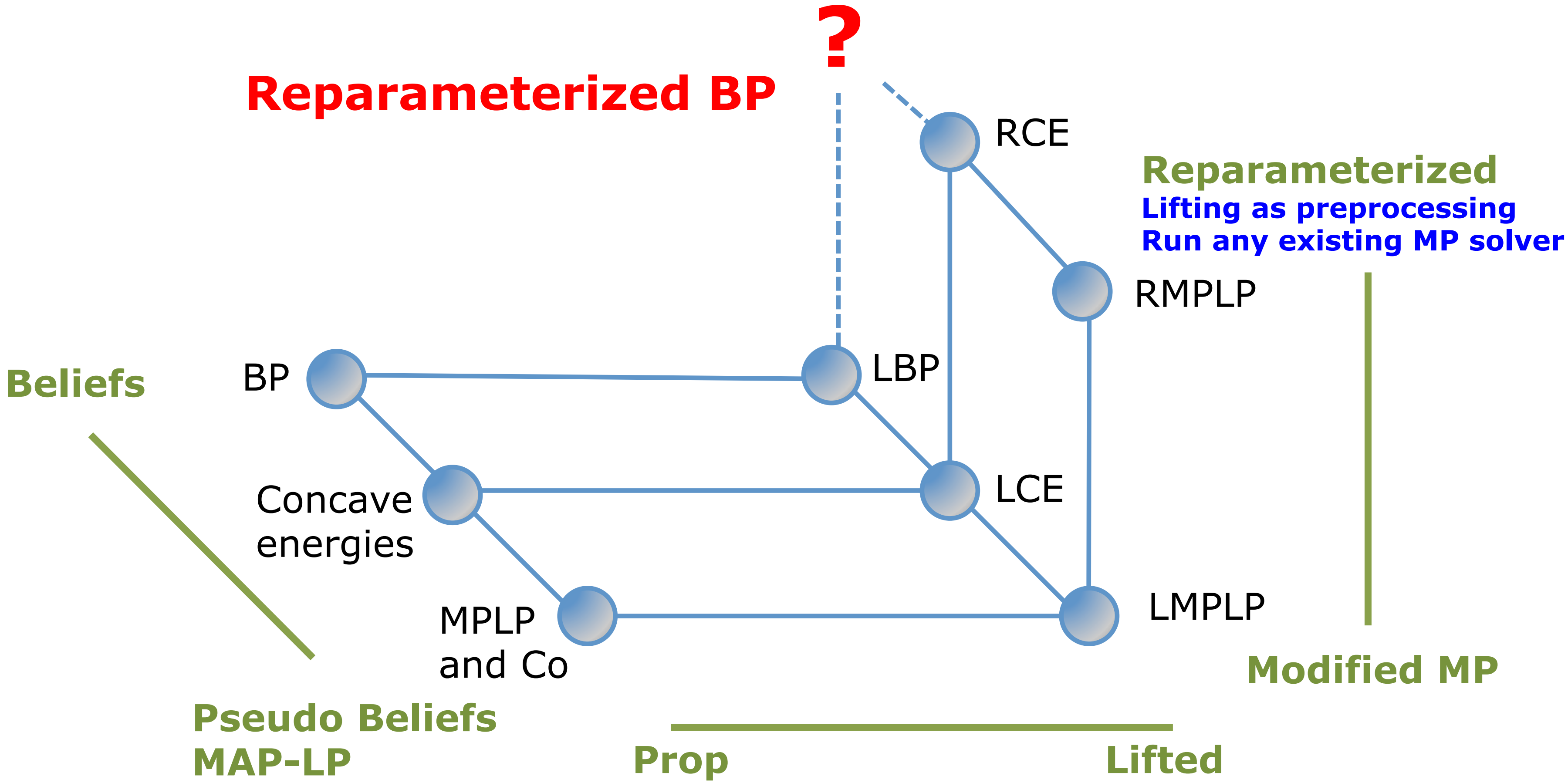


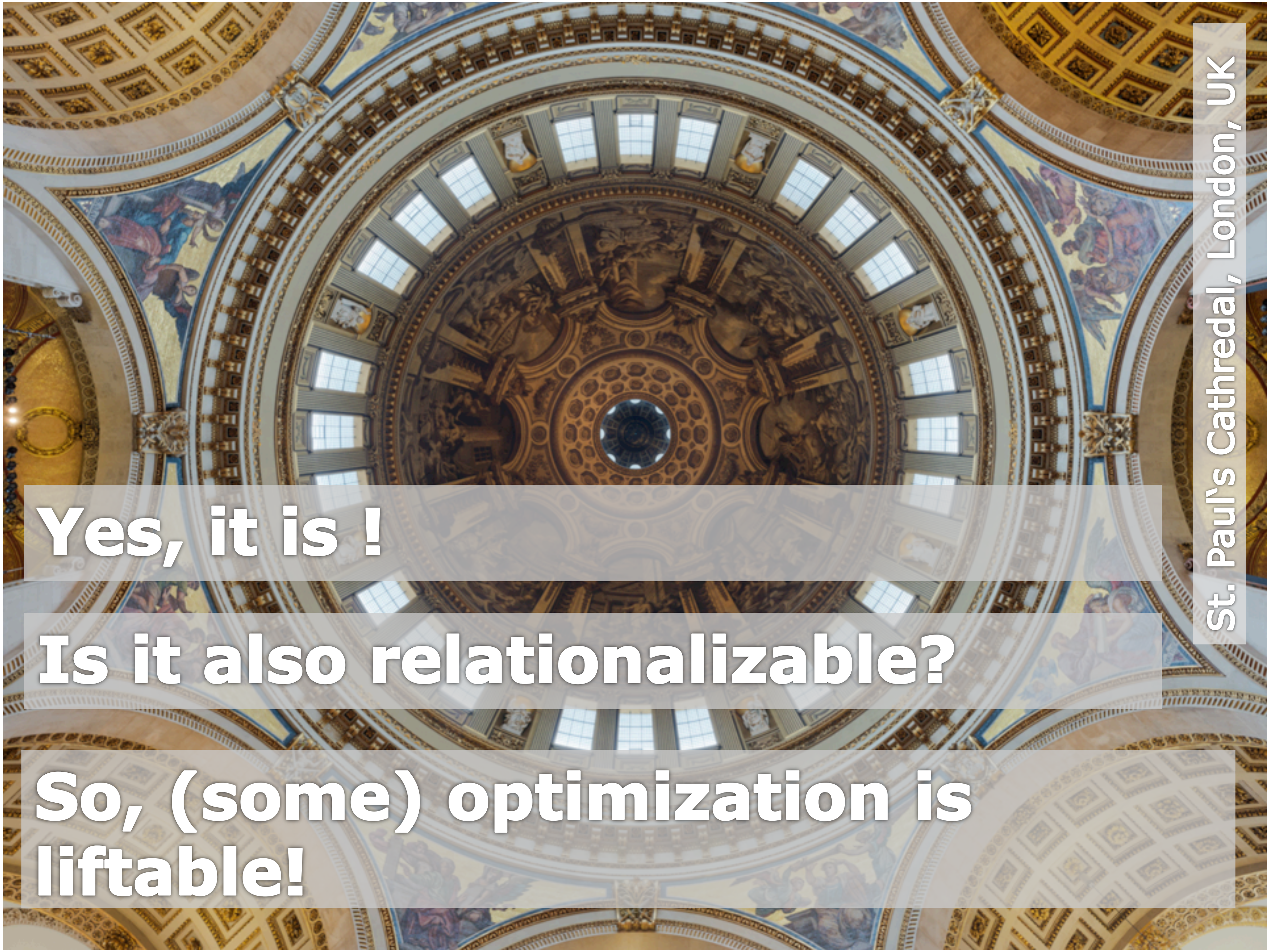
coloring

lifting

refine

Lifted inference =
Inference in a smaller, reparameterized model

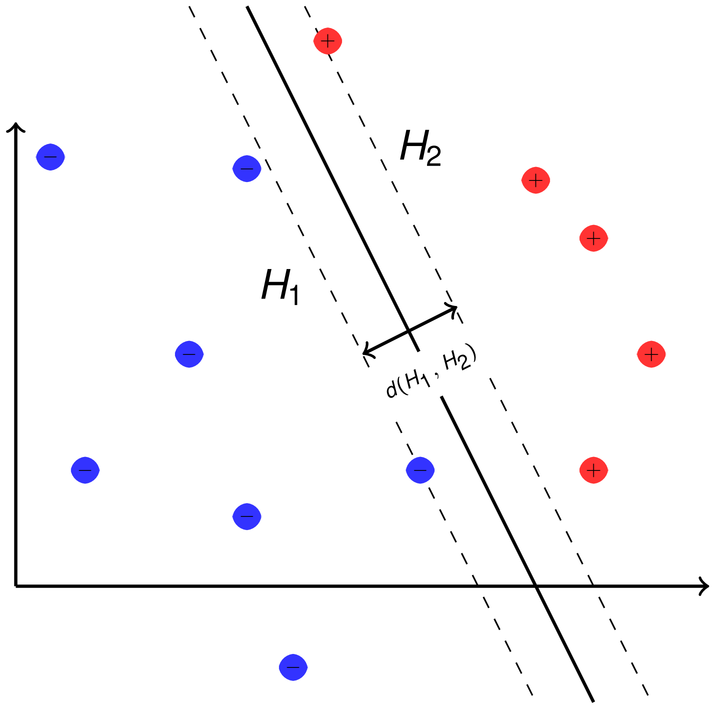



Yes, it is !
Is it also relationalizable?
So, (some) optimization is liftable!
St. Paul's Cathredal, London, UK

[Bennett 1999; Mangasarian 1999; Zhou, Zhang, Jiao 2002, ...]

Classification using LPs

$$H^* = \left\{ \vec{x} \mid \left\langle \vec{x}\,, \vec{\beta} \right\rangle + \beta_0 = 0 \right\}$$



$$d(H_1, H_2) = \frac{2}{||\vec{\beta}||}$$

Replace l_2- by l_1-, l_∞-norm in the standard SVM formulation

[Kersting, Mladenov, Tokmakov ARXIV 2014, AIJ 2015]

Declarative Machine Learning

Abstract LP-SVM

```
var pred/1;        #predicted label for unlabeled instances
var slack/1;       #th
var coslack/2;      #
var weight/1;      #th
var b/0;           #th
var r/0;           #margin

slack   = sum{label(I)} slack(I);
coslack = sum{cite(I1,I2),label(I1),query(I2)} slack(I1,I2)
        + sum{cite(I1,I2),label(I2),query(I1)} slack(I1,I2)

#find the largest margin. Here the C's encode trade-off parameters
minimize: -r + C(1) * slack + C(2) * coslack;
```

Logically parameterized LP variable (set of ground LP variables)

Logically parameterized LP objective

Logically parameterized LP constraint

```
#examples should be on the correct side of the hyperplane
subject to forall {I in label(I)}:
    label(I)*(innerProd(I) + b) + slack(I) >= r;
#weights are between -1 and 1
subject to forall {J in attribute(_, J)}: -1 <= weight(J) <= 1;
subject to : r >= 0;         #the margin is positive
subject to forall {I in label(I)}: slack(I) >= 0;    #slacks are positive
```

[Kersting, Mladenov, Tokmakov ARXIV 2014, AIJ 2015]

Declarative Machine Learning 2.0

Abstract LP-SVM

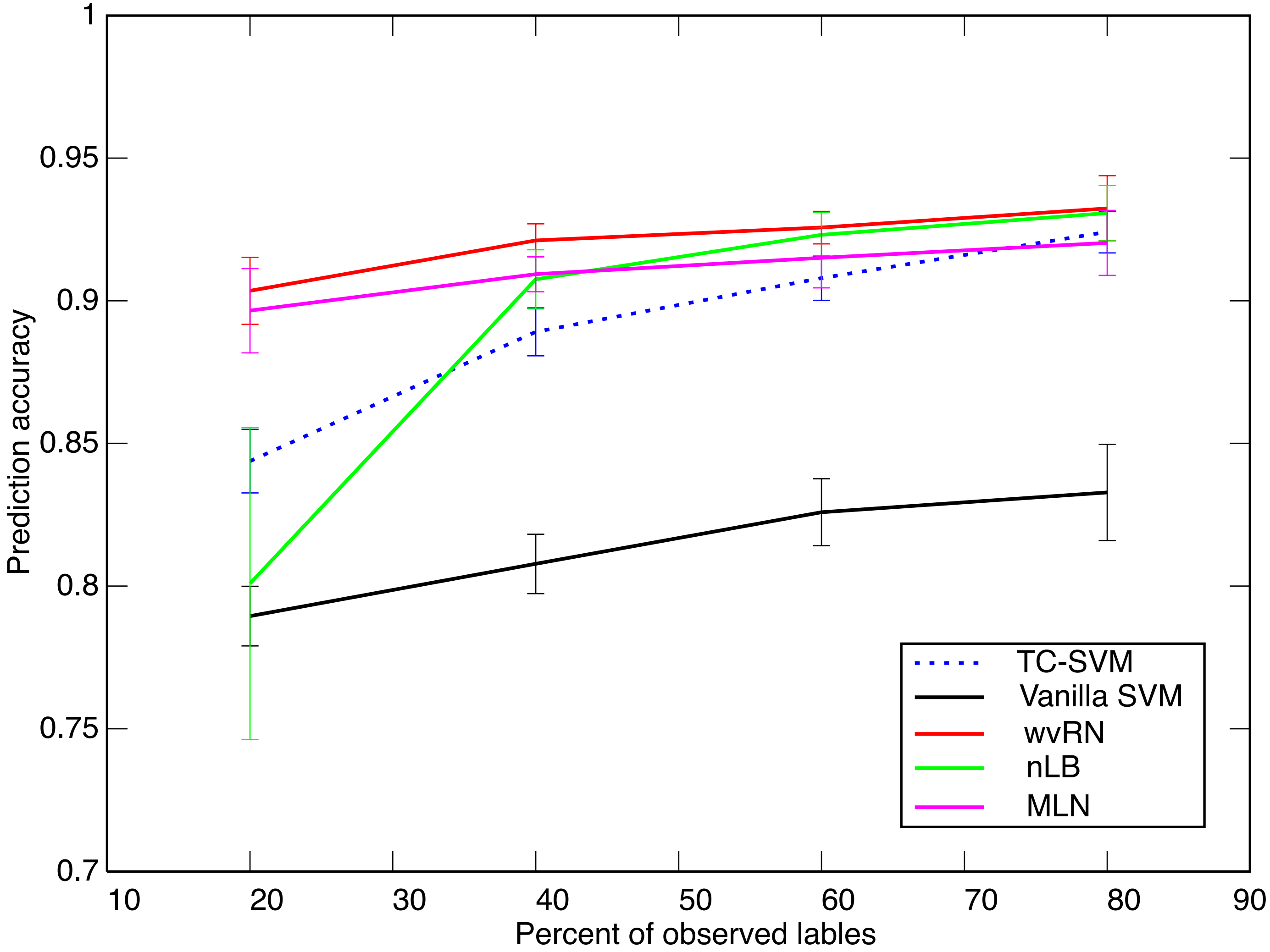
```
var pred/1;        #predicted label for unlabeled instances
var slack/1;       #the slacks
var coslack/2;       #slack between neighboring instances
var weight/1;      #the slope of the hyperplane
var b/0;           #the intercept of the hyperplane
var r/0;           #margin

slack   = sum{label(I)} slack(I);
coslack = sum{cite(I1,I2),label(I1),query(I2)} slack(I1,I2)
        + sum{cite(I1,I2),label(I2),query(I1)} slack(I1,I2)

#find the largest margin. Here the C's encode trade-off parameters
minimize: -r + C(1) * slack + C(2) * coslack;

subject to forall {I in query(I)}: pred(I) = innerProd(I) + b;
#related instances should have the same labels.
subject to forall {I1, I2 in cite(I1, I2), label(I1), query(I2)}
  label(I1) * pred(I2) +  slack(I1, I2) >= r;
#the symmetric case
subject to forall {I1, I2 in cite(I1, I2), label(I2), query(I1)}:
  label(I2) * pred(I1) + slack(I1, I2) >= r;

#examples should be on the correct side of the hyperplane
subject to forall {I in label(I)}:
    label(I)*(innerProd(I) + b) + slack(I) >= r;
#weights are between -1 and 1
subject to forall {J in attribute(_, J)}: -1 <= weight(J) <= 1;
subject to : r >= 0;          #the margin is positive
subject to forall {I in label(I)}: slack(I) >= 0;    #slacks are positive
```

Collective constraints

relational

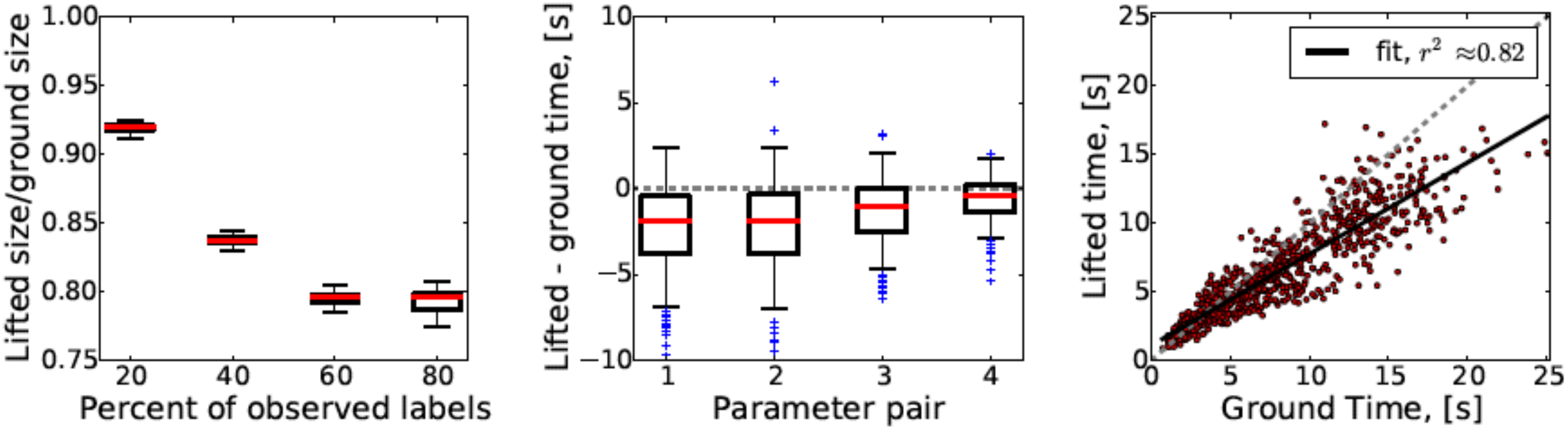
[Kersting, Mladenov, Tokmakov ARXIV 2014, AIJ 2015]

Cora (most common vs. rest)



technische universität dortmund

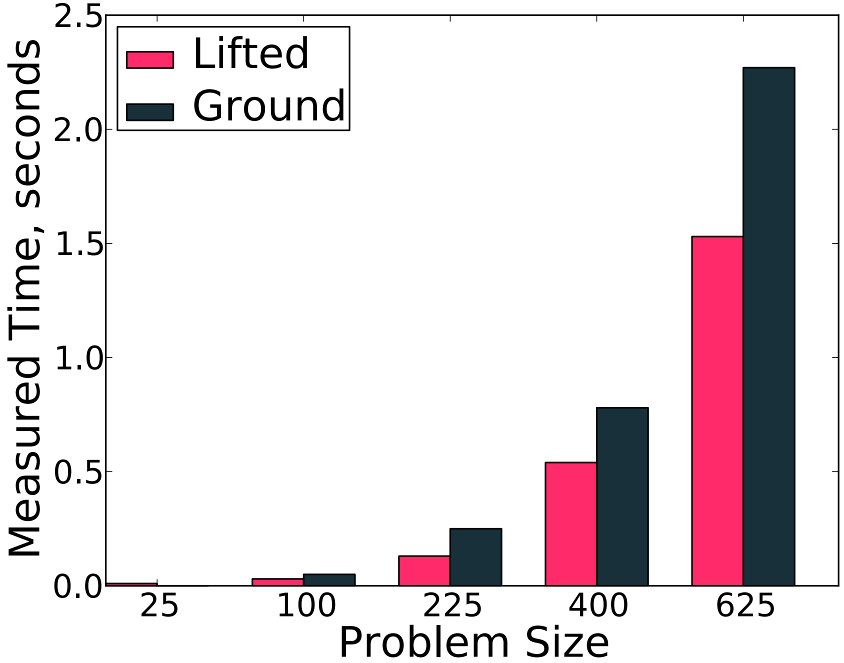
The more observed the more lifting
Faster end-to-end even in the light of Gurobi‘s fast pre-solving heuristics

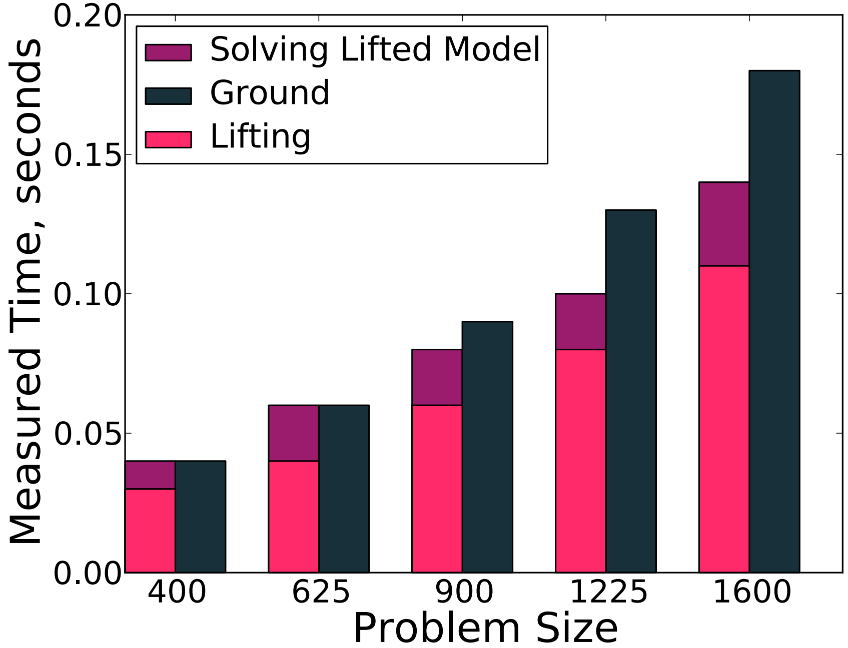


technische universität
dortmund

[Kersting, Mladenov, Tokmakov ARXIV 2014, AIJ 2015]

Relational MDP LPs





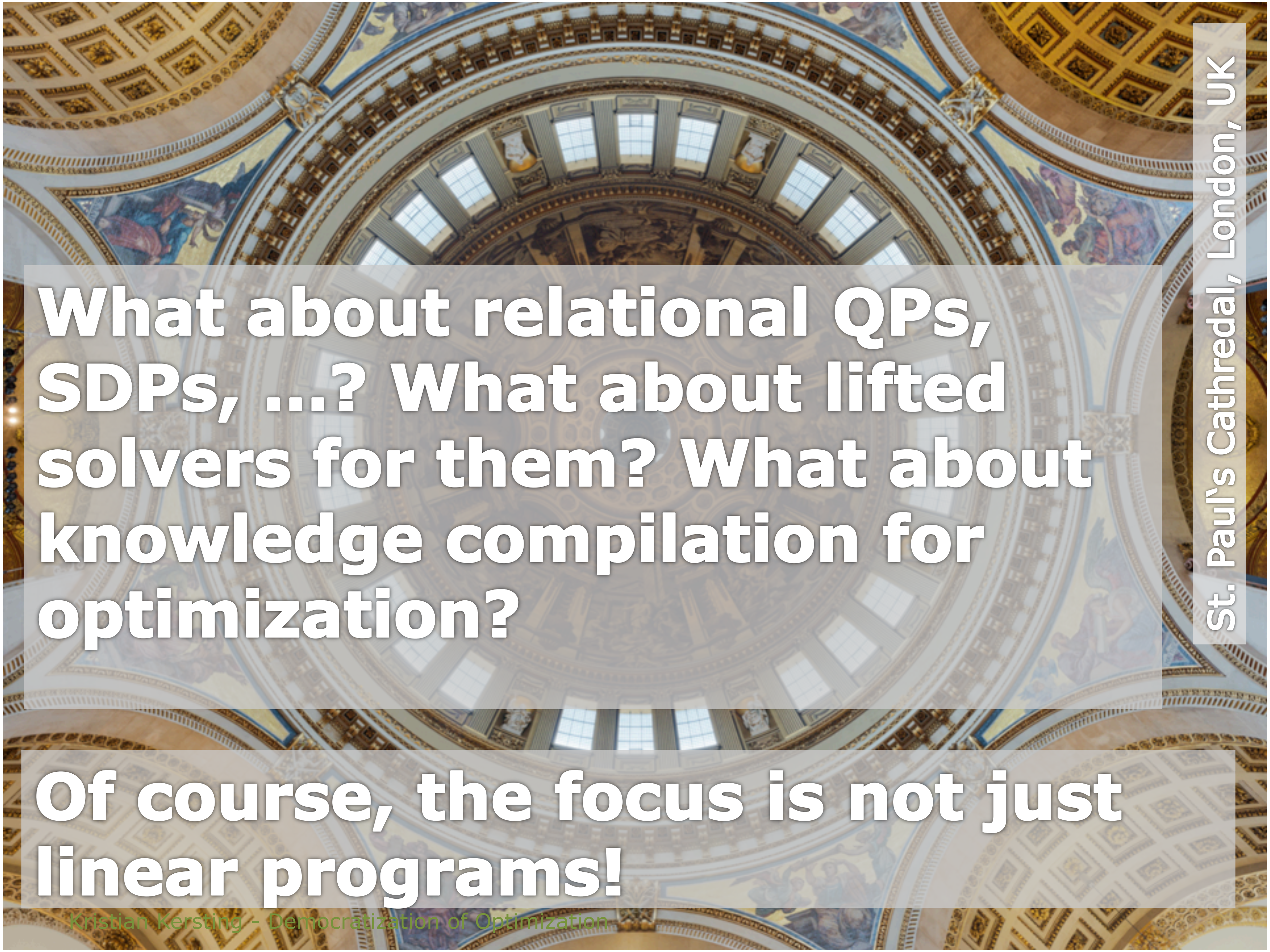
technische universität
dortmund

[Kersting, Mladenov, Tokmakov ARXIV 2014, AIJ 2015]

Relational MAP LPs

```
var m/2;  #single node, pairwise, and
var m/4;  #triplewise probabilities
var m/6;  #of configurations to be determined by the solver
#value of the MAP assignment
score = sum{w(P, V)} w(P, V) * m(P, V) +
  sum{w(P1, P2, V1, V2)} w(P1, P2, V1, V2) * m(P1, P2, V1, V2) +
  sum{w(P1, P2, P3, V1, V2, V3)} w(P1, P2, P3, V1, V2, V3) *
  m(P1, P2, P3, V1, V2, V3);

#marginalization of pairwise beliefs
marginalize(P1, P2, V1) = sum{w(P2, V2)} m(P1, P2, V1, V2);
...
#marginalization of ternary beliefs
marginalize(P1, P2, P3, V1) = sum{w(P3, V3), w(P2, V2)}
  m(P1, P2, P3, V1, V2, V3);
...
maximize: score;                          #find assignment with largest value
subject to forall {P in w(P, _)}:
    sum {w(P, V)} m(P, V) = 1;            #atom beliefs sum to one
#pairwise consistency constraints
subject to forall {P1, P2, V1 in w(P1, P2, V1, _)}:
    marginalize(P1, P2, V1) =  m(P1, V1);
...
#ternary consistency constraints
subject to forall {P1, P2, P3, V1 in w(P1, P2, P3, V1, _, _)}:
    marginalize(P1, P2, P3, V1) = m(P1, V1);
```

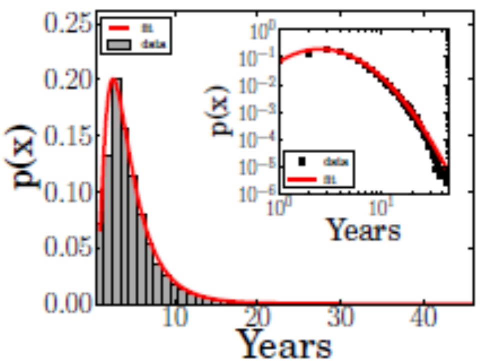
Stays fix for different relational probabilistic model, which are specified in a knowledge base


What about relational QPs, SDPs, ...? What about lifted solvers for them? What about knowledge compilation for optimization?
Of course, the focus is not just linear programs!
St. Paul's Cathredal, London, UK

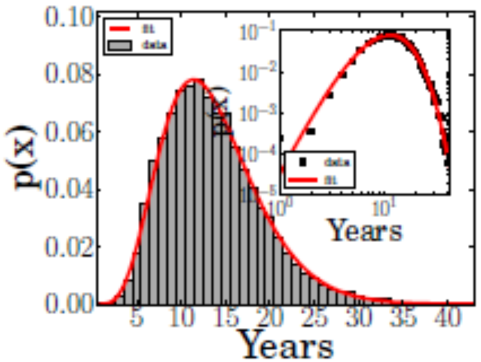
[Hadiji, Mladenov, Bauckhage, Kersting IJCAI 2015]

Relational and Compressed Label Propagation

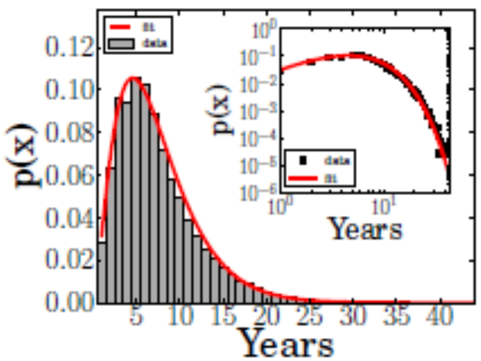
- **Many human activities have been shown to exhibit universal patterns.**
- **What about researcher migration in CS?**
- **Inferred from 1 million authors of 1,9 million papers**



Early Stage: LogNormal



Later Stages: Gamma



Brain Circulation: Gamma

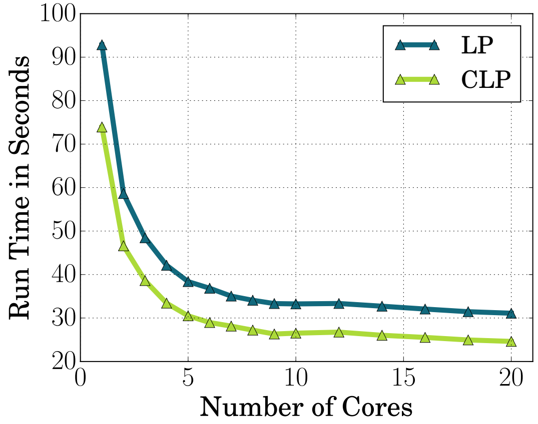
[Hadiji, Mladenov, Bauckhage, Kersting IJCAI 2015]

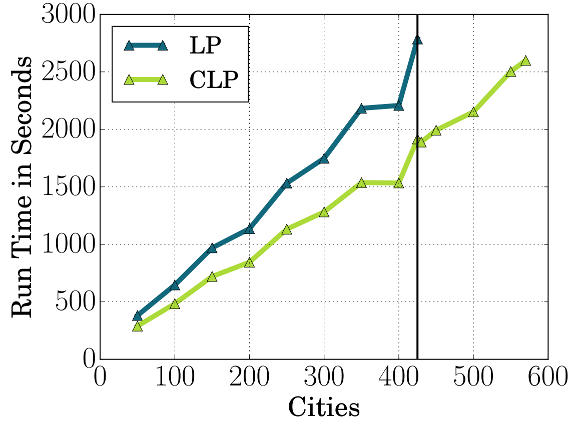
Relational and Compressed Label Propagation

Relational way to specify Label Propagation matrix

$$w_{ij} = w_{ij} + \lambda_2 \text{ if } \mathrm{a}(i) = \mathrm{a}(j) \wedge \mathrm{y}(i) = \mathrm{y}(j)$$

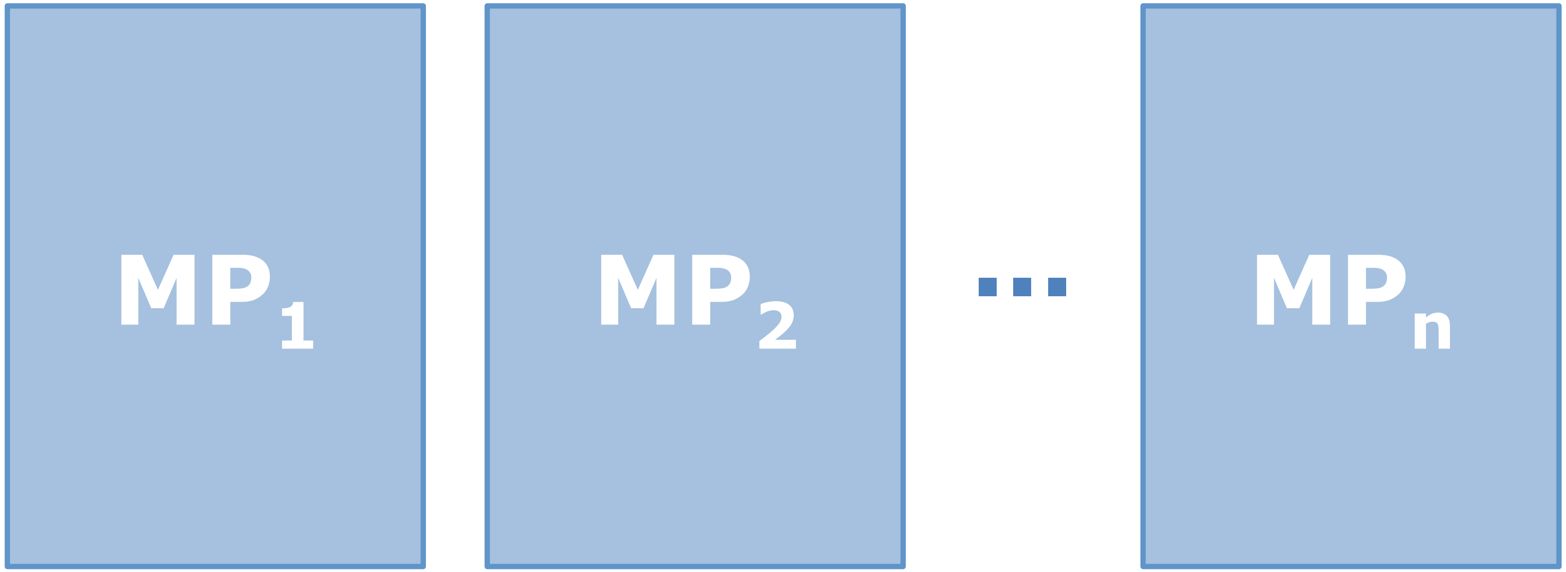
Fractional automorphims for compressing the resulting label propagation matrix; use any label propagation approach (even QP once) on the compressed matrix





[Kersting, Mladenov, Tokmakov ARXIV 2014, AIJ 2015]

Relational Mathematical Programming



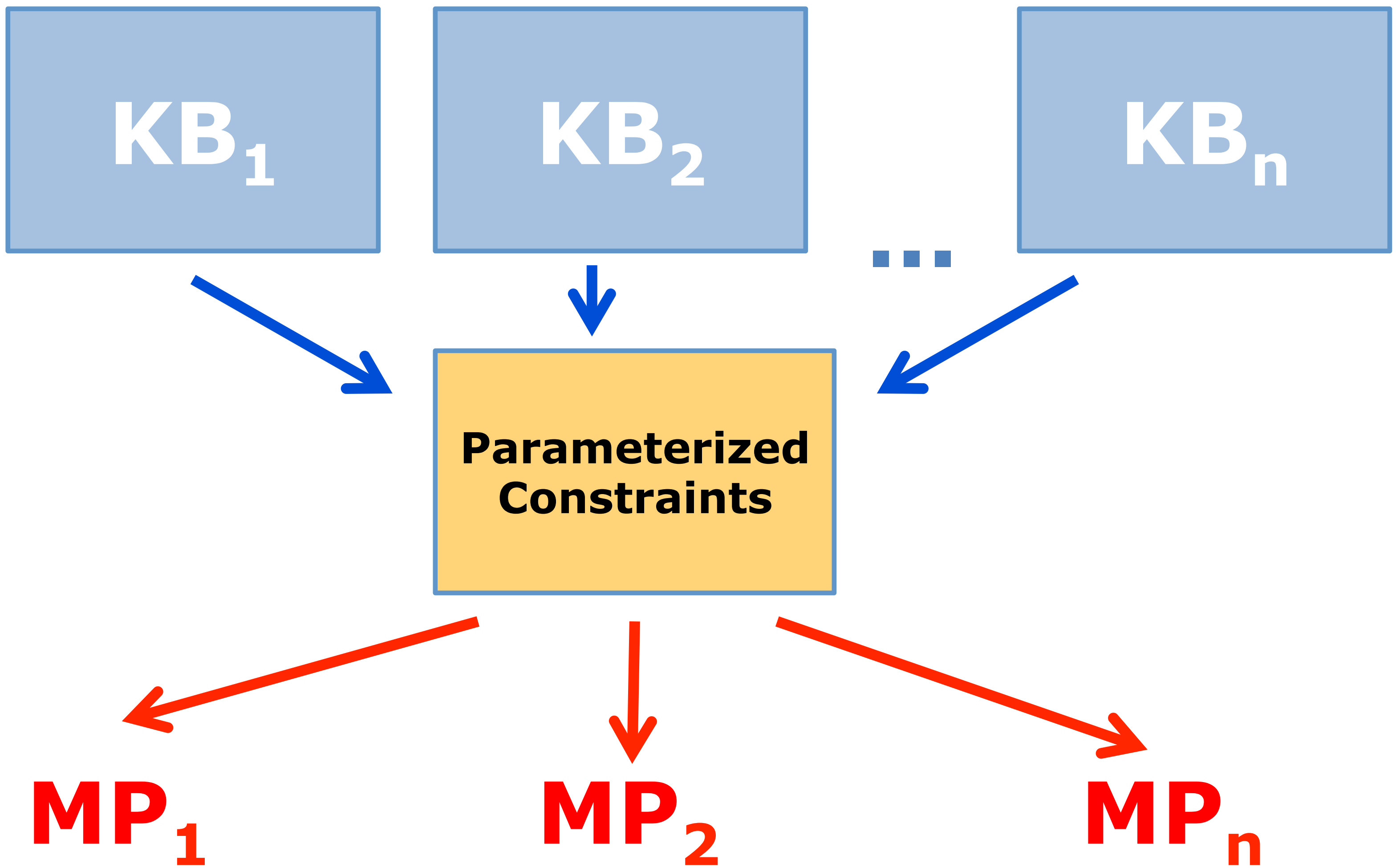
[Kersting, Mladenov, Tokmakov ARXIV 2014, AIJ 2015]

Relational Mathematical Programming

KB_1	KB_2	…	KB_3
Constraints_1	Constraints_2		Constraints_3

[Kersting, Mladenov, Tokmakov ARXIV 2014, AIJ 2015]

Relational Mathematical Programming



However, relational programming is not the answer to everything

Let`s embed relational mathematical programming into an imparative programming

reloop

http://www-ai.cs.uni-dortmund.de/weblab/static/RLP/html/

RELOOP: A Toolkit for Relational Convex Optimization

```
model = RlpProblem("flow", LpMaximize, PyDatalogLogKb(), Pulp)
```

```
flow = numeric_predicate("flow", 2)
cap = numeric_predicate("cap", 2)
```

```
model.add_reloop_variable(flow)
```

```
model += RlpSum([X, Y], source(X) & e
```

```
outFlow = RlpSum([X, ], edge(X, Z), f
inFlow = RlpSum([Y, ], edge(Z, Y), fl

model += ForAll([Z, ], node(Z) & ~source(Z) & ~target(Z), inFlow |eq| outFlow)
```

...

Using a probabilistic programming language we can even get stochastic RMPs

Loops and relations get interwined, and models can refer to each other

Take-away Messages

1. Graphical models allow to deal with uncertainty
2. Graphs/Matrices are not enough, we need logic /high-level languages
3. Tree-Width is not the end of the story
4. Probabilities are not enough we need optimization
5. Relations and loops should go together

technische universität dortmund

Democratization of Optimization

- Reduce the level of expertise necessary to build optimization applications
- Shorten mathematical program code to make models faster to write and easier to communicate
- Reduce development time and cost to encourage experimentation
- Facilitate the construction of more sophisticated models that incorporated rich domain knowledge
- Speed up solvers by exploiting language properties, compression, and compilation

Achieving this requires the help of all of you! ML, KR, CP, SAT, PL, ...

Thanks for you attention

technische universität
dortmund