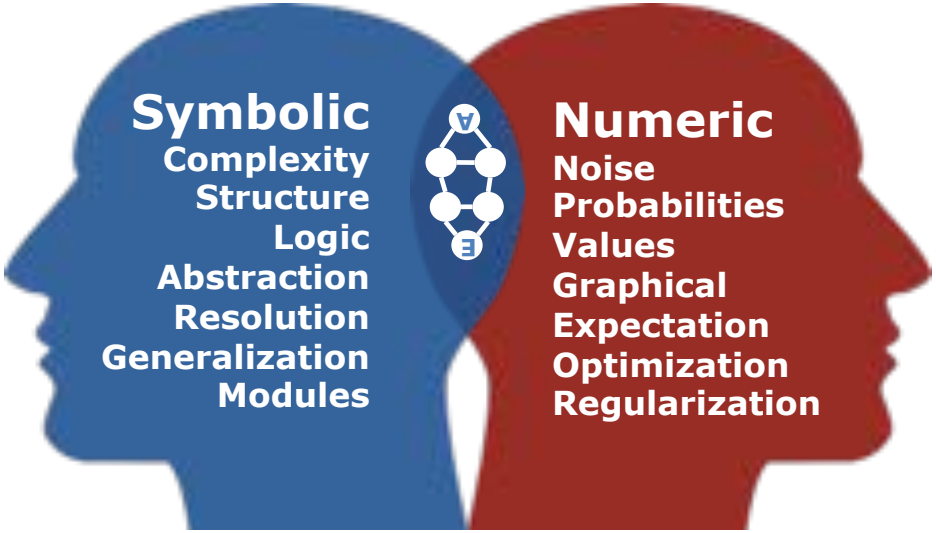
Kristian Kersting

Lifted Statistical Machine Learning

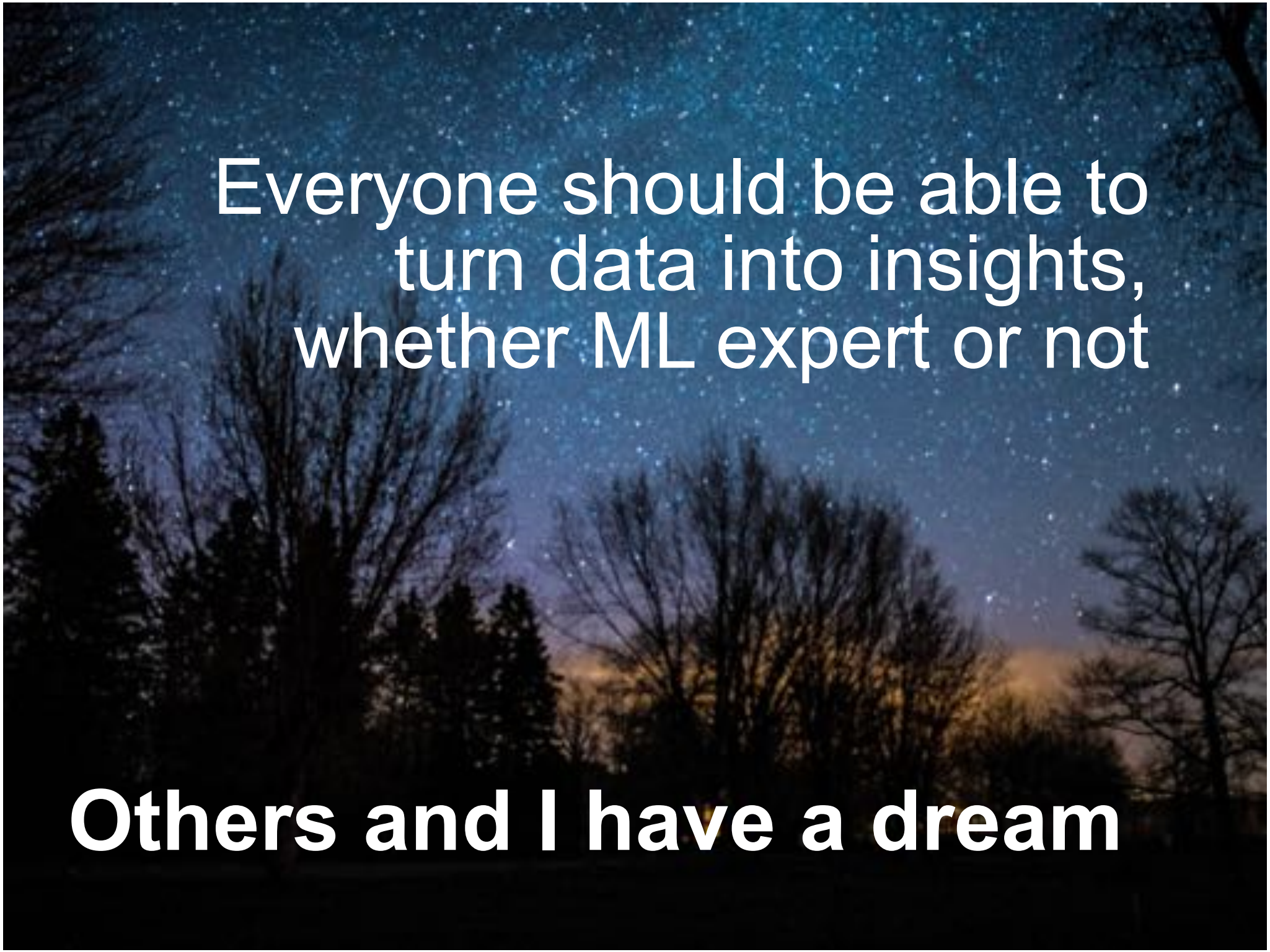
Computational modeling of complex AI systems that learn and think

TECHNISCHE UNIVERSITÄT DARMSTADT
Centre for Cognitive Science
Fachbereich Informatik

Thanks to Babak Ahmadi, Vincent Conitzer, Rina Dechter, Luc De Raedt, Pedro Domingos, Peter Flach, Dieter Fensel, Florian Ficher, Vibhav Gogate, Carlos Guestrin, Daphen Koller, Nir Friedman, Martin Mladenov, Ray Mooney, Sriraam Natarajan, David Poole, Fabrizio Riguzzi, Dan Suciu, Guy van den Broeck, and many others for making their slides publically available



ACAI 2018
Summer School on Statistical Relational Artificial Intelligence
August 27th - 31st 2018, Ferrara, Italy
EurAI
Part of Relational Artificial Intelligence Days (RAID)


Everyone should be able to
turn data into insights,
whether ML expert or not
Others and I have a dream

This poses many deep and fascinating questions

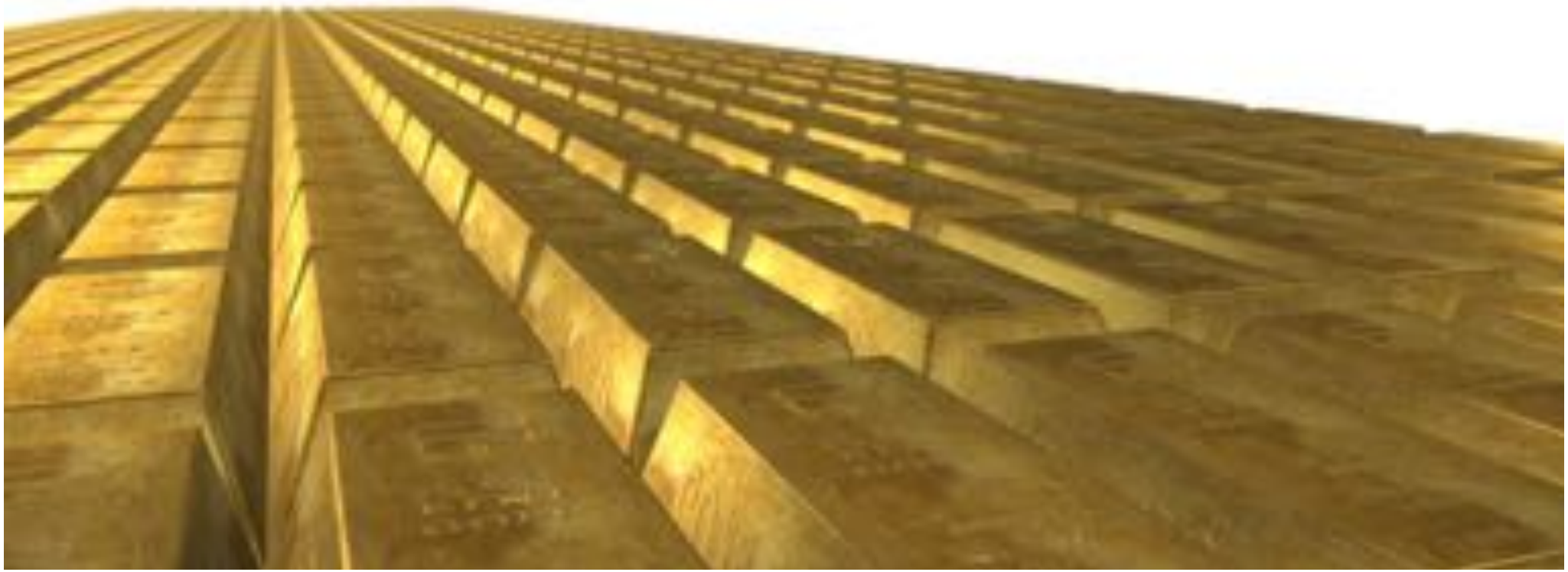
How can we make ML and AI more tractable?

How can computers reason about and learn with complex data and knowledge?

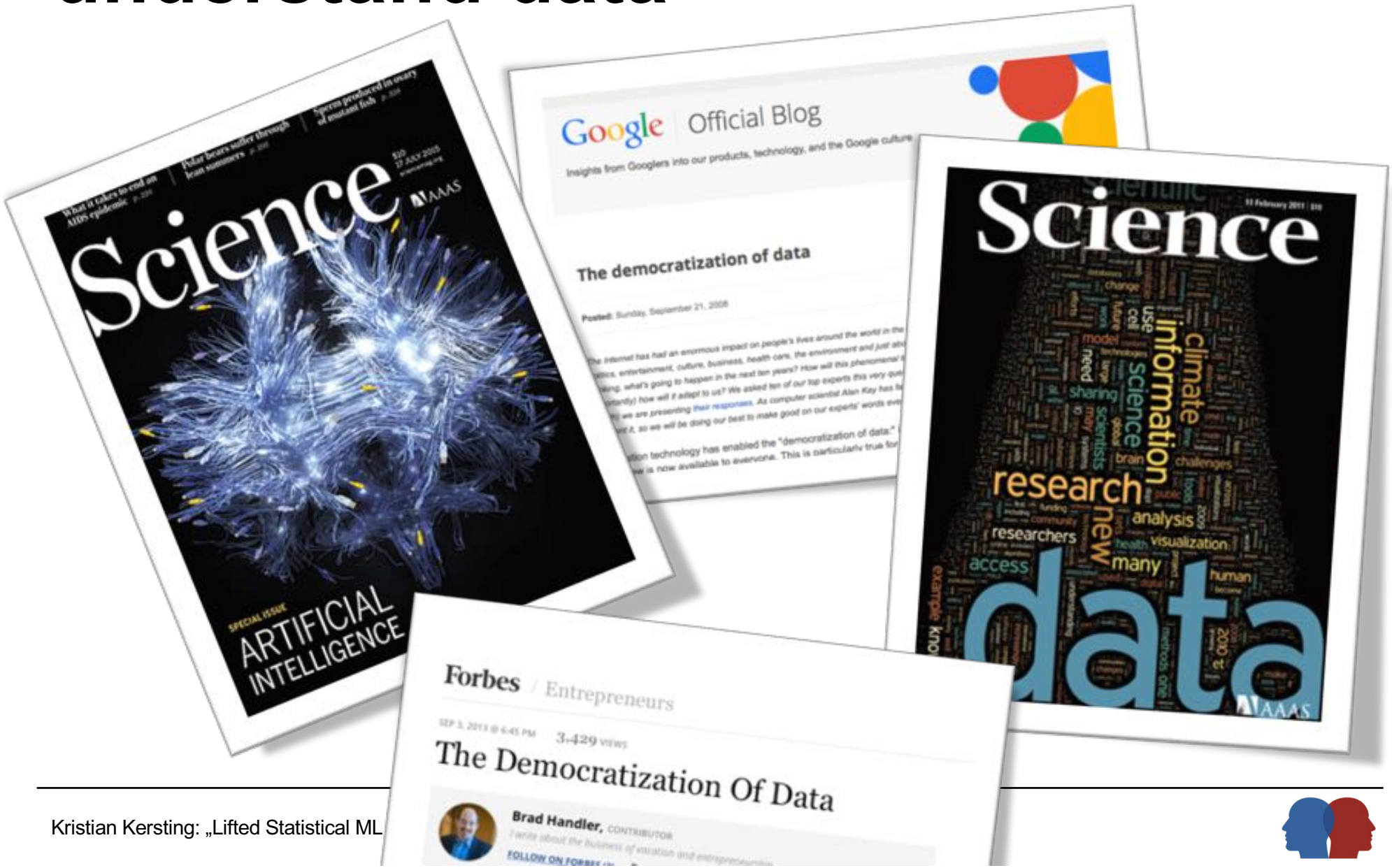
How can computers decide autonomously which representation and algorithms are best for the data/problem?

How can computers understand data with minimal expert input?

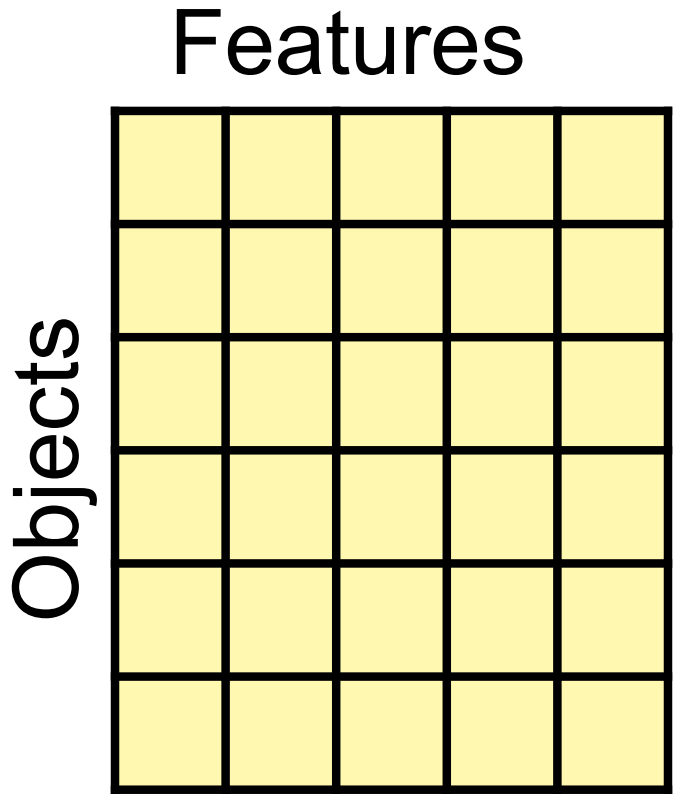
Today is the golden era of data



Arms race to deeply understand data

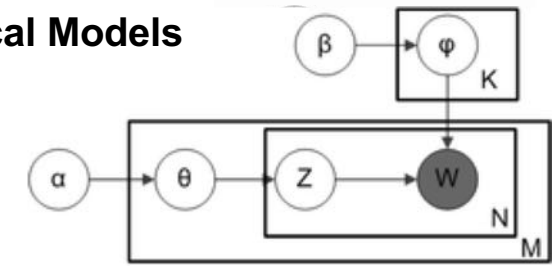


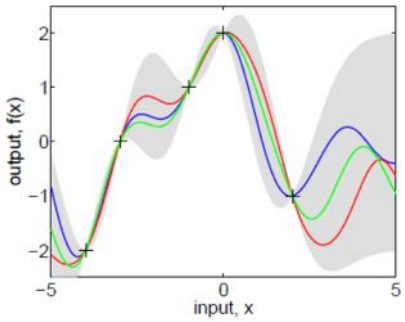
Bottom line:
Take your data spreadsheet …



… and apply Machine Learning

Probabilistic Graphical Models
Arithmetic Circuits





Gaussian Processes

Features

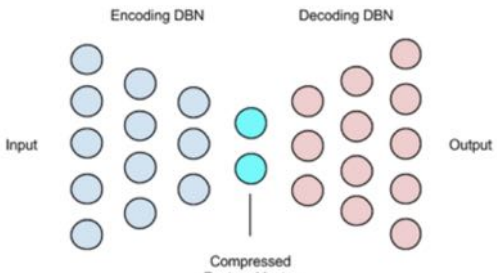
Objects

Is it really that simple?

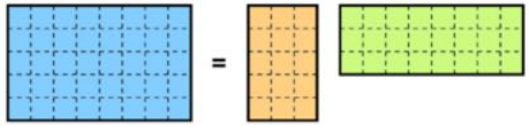
Big Model

Small Model

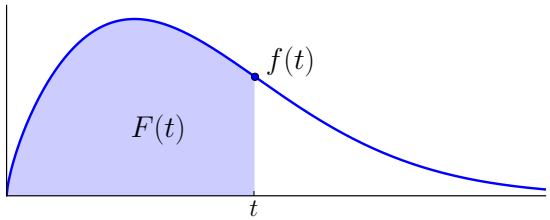
Distillation/LUPI



Autoencoder,
Deep Learning



Big Data Matrix Factorization



Diffusion Models

and many more …

Mind the **data science loop**

Research question

Data collection and preparation

Continuous? Discrete? Categorial? …

Multinomial? Gaussian? Poisson? ...

ML

Discuss results

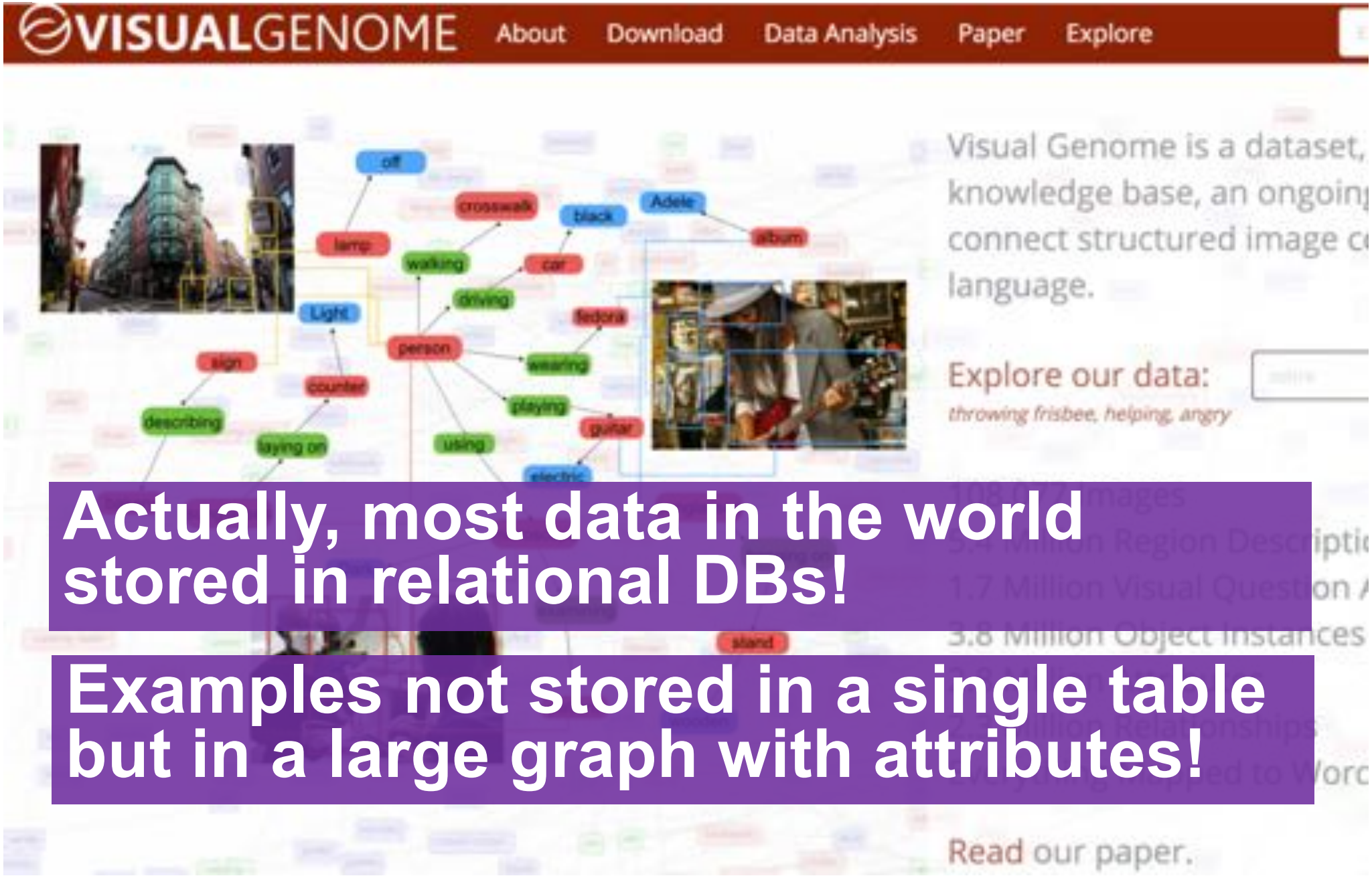
How to report results? What is interesting?

Answer found?

Deployment

Complex data networks abound

[Lu, Krishna, Bernstein, Fei-Fei „Visual Relationship Detection“ CVPR 2016]



Actually, most data in the world stored in relational DBs!

Examples not stored in a single table but in a large graph with attributes!

We have to **democratize AI, Machine Learning, and Data Science**

We have to work on **Systems AI**, so that we know how to rapidly combine, deploy, and maintain algorithms

So yes, today is the golden era of data …

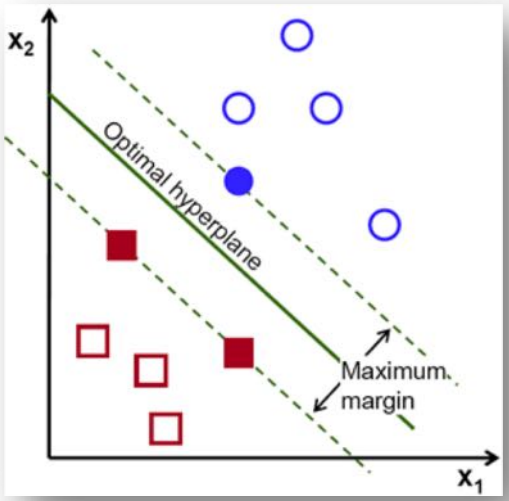
… for the best-trained, best-funded Machine Learning and Artificial Intelligence teams

Let’s say we want to classify publications into scientific disciplines

$$\min_{\mathbf{w},b,\boldsymbol{\xi}} \mathcal{P}(\mathbf{w},b,\boldsymbol{\xi}) = \frac{1}{2}\mathbf{w}^2 + C\sum_{i=1}^{n}\xi_i$$

$$\text{subject to} \quad \begin{cases} \forall i & y_i(\mathbf{w}^\top \Phi(\mathbf{x}_i) + b) \geq 1 - \xi_i \\ \forall i & \xi_i \geq 0 \end{cases}$$

Mathematics is not a high-level programming language!



Support Vector Machines
Cortes, Vapnik MLJ 20(3):273-297, 1995

Machine Learning Programming

TECHNISCHE UNIVERSITÄT DARMSTADT

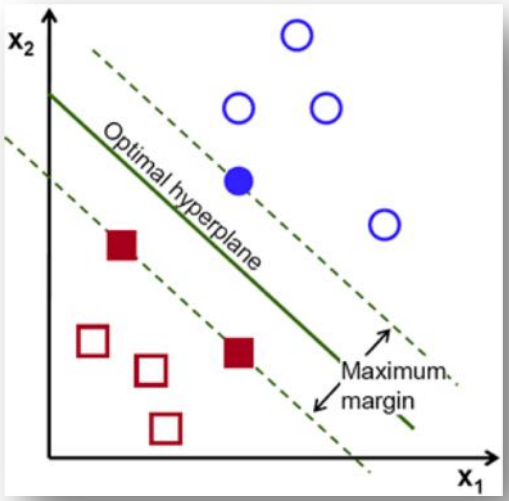
Write down SVM in „paper form.“ The machine compiles it into solver form.

```
#QUADRATIC OBJECTIVE
minimize: sum{J in feature(I,J)} weight(J)**2 + c1 * slack + c2 * coslack;

#labeled examples should be on the correct side
subject to forall {I in labeled(I)}: labeled(I)*predict(I) >= 1 - slack(I);

#slacks are positive
subject to forall {I in labeled(I)}: slack(I) >= 0;
```

Embedded within Python s.t. loops and rules can be used

reloop

RELOOP: A Toolkit for Relational Convex Optimization



Support Vector Machines
Cortes, Vapnik MLJ 20(3):273-297, 1995

But wait, publications are citing each other. OMG, I have to use graph kernels!

REALLY?

No, just add two lines of code!

TECHNISCHE UNIVERSITÄT DARMSTADT

Write down SVM in „paper form.“ The machine compiles it into solver form.

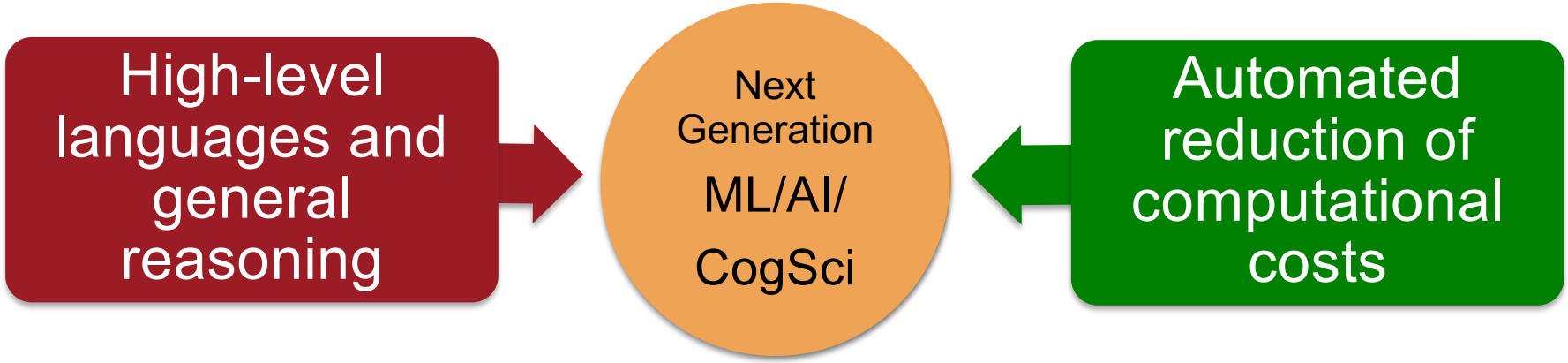
```
#QUADRATIC OBJECTIVE
minimize: sum{J in feature(I,J)} weight(J)**2 + c1 * slack + c2 * coslack;

#labeled examples should be on the correct side
subject to forall {I in labeled(I)}: labeled(I)*predict(I) >= 1 - slack(I);

#slacks are positive
subject to forall {I in labeled(I)}: slack(I) >= 0;

#TRANSDUCTIVE PART
#cited instances should have the same labels.
subject to forall {I1, I2 in linked(I1, I2)}: labeled(I1) * predict(I2) >= 1 - slack(I1, I2)
subject to forall {I1, I2 in linked(I1, I2)}: coslack(I1, I2) >= 0; #coslacks are positive
```

Citing papers share topics

No kernel, the structure is expressed within the constraints!

Take-away message

To move beyond deep learning, ML, AI and Computational Cognitive Science need a crossover with data and programming abstractions as well as general reasoning



- High-level languages increase the number of people who can successfully build ML/AI applications that involve learning and reasoning
- To deal with the computational complexity, we need ways to automatically reduce the solver costs

Roadmap of this tutorial

TECHNISCHE UNIVERSITÄT DARMSTADT

Two parts

1. Lifted Probabilistic Inference and Tractable Probabilistic Models
2. Statistical Relational Learning and Probabilistic Programming

- High-level languages increase the number of people who can successfully build ML/AI applications that involve learning and reasoning
- To deal with the computational complexity, we need ways to automatically reduce the solver costs

Lifted Statistical Machine Learning

Computational modeling of complex AI systems that learn and think

TECHNISCHE UNIVERSITÄT DARMSTADT

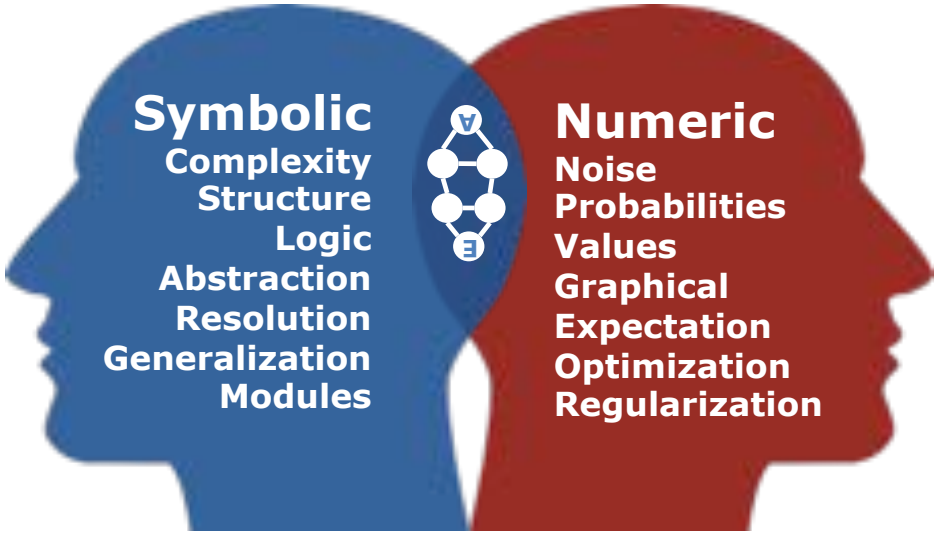
Centre for Cognitive Science

Fachbereich Informatik

Part I: Tractable Probabilistic Models

Kristian Kersting

Thanks to Babak Ahmadi, Vincent Conitzer, Rina Dechter, Luc De Raedt, Pedro Domingos, Peter Flach, Dieter Fensel, Florian Ficher, Vibhav Gogate, Carlos Guestrin, Daphen Koller, Nir Friedman, Martin Mladenov, Ray Mooney, Sriraam Natarajan, David Poole, Fabrizio Riguzzi, Dan Suciu, Guy van den Broeck, and many others for making their slides publically available



Judea Pearl received 2012 the ACM Turing Award 2012 for his work on graphical models

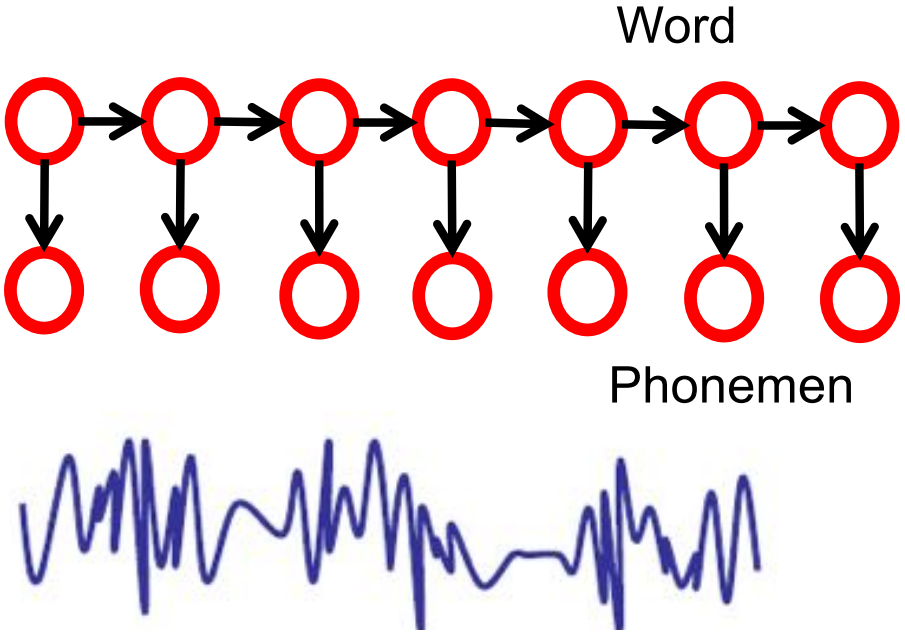
Roadmap of Part I

1 Basics of graphical models

2a Exploiting symmetries for inference

2b Deep probabilistic models

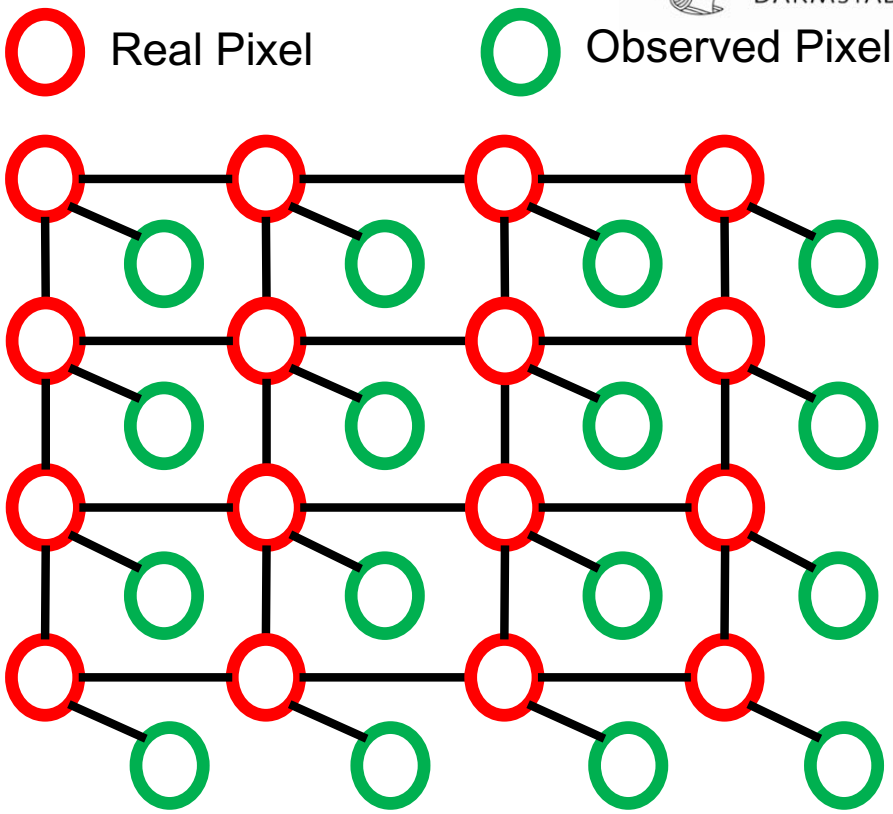
Application: Speech recognition



„He ate the cookies on the couch“

Infer the words from spoken language:
Hidden Markov Model

Application: Image Denoising



Infer the original image from the noisy observed one: Markov networks

Graphical models are omnipresent

Information retrivel, search, collaborative filtering, gene expression analysis, natural language processing, bioinformatics, and many,

many,

many,

many

more!

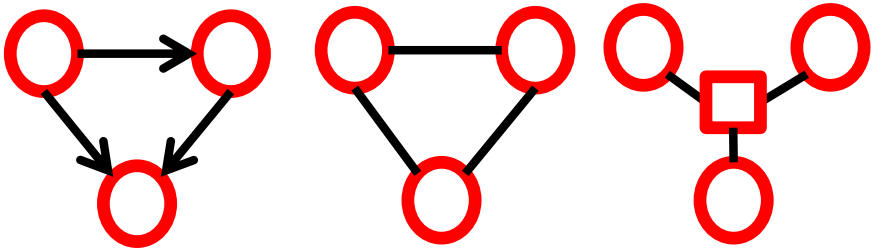
OK, so what are probabilities and graphical models?

In a Nutshell, Graphical Models are …

Probability

… a graphical notation for (conditional) independency assumptions and therefore a (hopefully) compact specification of probability distributions

Nodes=
Random Variables (RVS)
Edges=
Dependencies among RVs



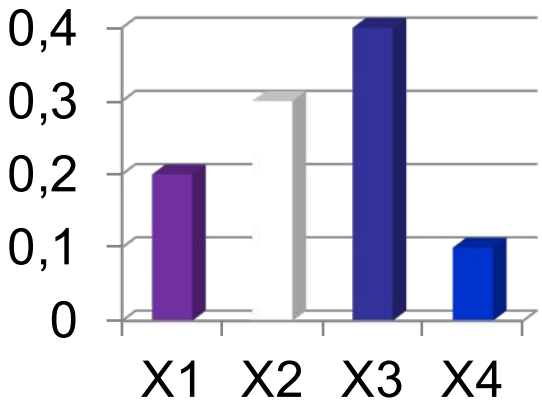
Bayesian Networks | Markov Networks | Factor Graphs

Discrete Random Variables

Finite set X of possible states

$$X \in \{x_1, x_2, x_3, \ldots, x_n\}$$

$$P(x_i) \geq 0 \quad \sum_{i=1}^{n} P(x_i) = 1$$



OK, but answering the question requires the joint distribution

What is the probability that smoking causes cancer?

Smoking

no	few	many
0.800	0.150	0.050

Cancer

no	benigne	maligne
0.935	0.046	0.019

Joint Distribution

Probability that X=x and Y=y are “true”

$$P(x, y) \equiv P(X = x \wedge Y = y)$$

Cancer

Smoking	no	benigne	maligne
no	0.768	0.024	0.008
few	0.132	0.012	0.006
many	0.035	0.010	0.005

The joint distribution allows us to answer any question! But how?

Make use of basic probability theory

TECHNISCHE UNIVERSITÄT DARMSTADT

Marginalization

$$P(Y) = \sum_{i=1}^{n} P(Y, x_i)$$

sum

Cancer

Smoking	No	Benigne	Maligne	TOTAL
No	0.768	0.024	0.008	0.800
few	0.132	0.012	0.006	0.150
many	0.035	0.010	0.005	0.050
TOTAL	0.935	0.046	0.019	

P(smoking)

P(cancer)

Product rule & **conditional probability**

$$P(X,Y) = \boxed{P(X \mid Y)} P(Y) = P(Y \mid X) P(X)$$

Probability that X=x if we have observed Y=y (P(y)>0)

Probably the most important rule: Bayes

$$P(R,K) = P(R \mid K)P(K) = P(K \mid R)P(R)$$

$$\boxed{P(R \mid K) = \frac{P(K \mid R)P(R)}{P(K)} = \frac{P(K,R)}{P(K)}}$$

cancer

smoking	no	benigne	maligne
no	0.768	0.024	0.008
few	0.132	0.012	0.006
many	0.035	0.010	0.005
TOTAL	0.935	0.046	0.019

P(Krebs)

Since we know already P(R,K) und auch P(K), just divide them.

Probably the most important rule: Bayes

$$P(R,K) = P(R \mid K)P(K) = P(K \mid R)P(R)$$

$$\boxed{P(R \mid K) = \frac{P(K \mid R)P(R)}{P(K)} = \frac{P(K,R)}{P(K)}}$$

cancer

smoking	no	benigne	maligne
no	0.768/0.935	0.024/ 0.46	0.008/ 0.019
few	0.132/0.935	0.012/ 0.46	0.006/ 0.019
many	0.035/0.935	0.010/ 0.46	0.005/ 0.019
TOTAL	0.935	0.046	0.019

P(cancer)

Probably the most important rule: Bayes

$$P(R,K) = P(R \mid K)P(K) = P(K \mid R)P(R)$$

$$\boxed{P(R \mid K) = \frac{P(K \mid R)P(R)}{P(K)} = \frac{P(K,R)}{P(K)}}$$

cancer= …)

P(smoking|

	no	benigne	maligne
no	0.821	0.522	0.421
few	0.141	0.261	0.316
many	0.037	0.217	0.263

As long as all entries are >0, everything can be computed! Mission completed?

No!! Our mission has just started

Joint distribution is enumerating everything

- Worst-case run time: $O(2^n)$
  - n = # of RVs
- Space is $O(2^n)$ too
  - Size of the table of the joint distribution

Main idea: make use of independencies to compress the representation

Indpendency

(Current) age and the gender of a person are independent

$$P(G, A) = P(G) \cdot P(A)$$

$$P(A \mid G) = P(A)$$

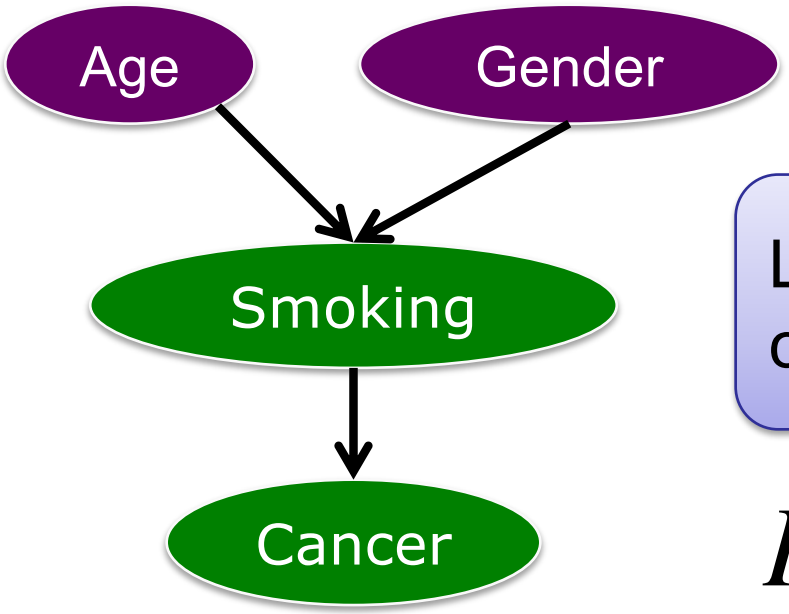
$$P(G \mid A) = P(G)$$

You would not give me money for information on the gender to know the age of a person!

Conditional Independence

Cance is independent of age and gender, if the person smokes.

If you have not observed anything,age and gender are independent.



Less entries, therefore lower complexity

$$P(C \mid S, G, A) = P(C \mid S)$$

Bayesian Networks [Pearl 1989]

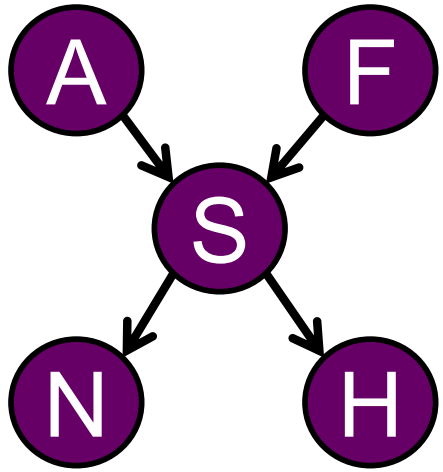
Set of random variables $\{X_1,\ldots,X_n\}$

Directed, acyclic graph (DAG)

To each RV Xi we associate the

conditional probabilitiy distribution:

$$P(X_i \mid \mathrm{Pa}(X_i))$$



The **joint distribution** is

$$P(X_1,\ldots,X_n) = \boxed{\prod_{i=1}^{n} P(X_i \mid \mathrm{Pa}(X_i))}$$

BN semantics

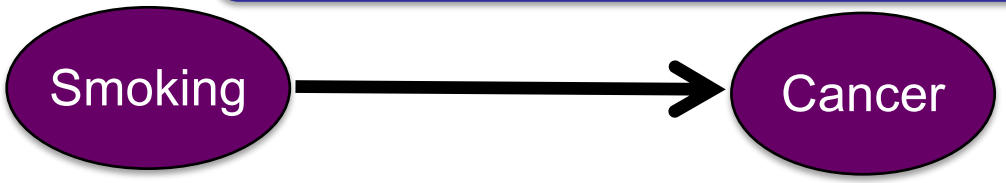
Local Markov Assumption

Each RV X is independent of ist „non-descendant“ given ist parents (Xi ⊥ nonDescendants| Pa_{Xi})

Example

But how do we do inference?

$R \in \{no, few, many\}$



P(S=n)	0.80
P(S=f)	0.15
P(S=m)	0.05

$K \in \{no, benigne, maligne\}$

Smoking=	n	f	m
P(C=n)	0.96	0.88	0.60
P(C=b)	0.03	0.08	0.25
P(C=m)	0.01	0.04	0.15

What is Inference?

Query: $P(X \mid e)$

Definiton of conditional probability $P(X \mid e) = \frac{P(X, e)}{P(e)}$

Up to normalization $P(X \mid e) \propto P(X, e)$

Hence, this rewrites to

$$P(\mathbf{Y}) = \sum_{X_i \notin \mathbf{Y}} \prod_{i=1}^{n} P(X_i \mid \mathrm{Pa}(X_i))$$

BN semantics

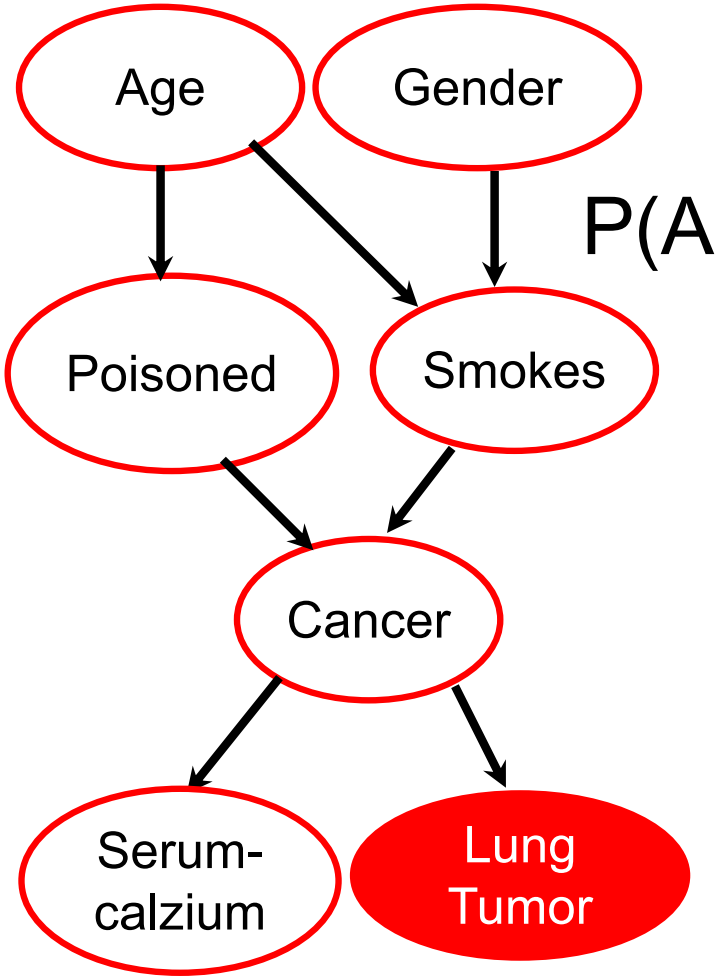
Marginalization

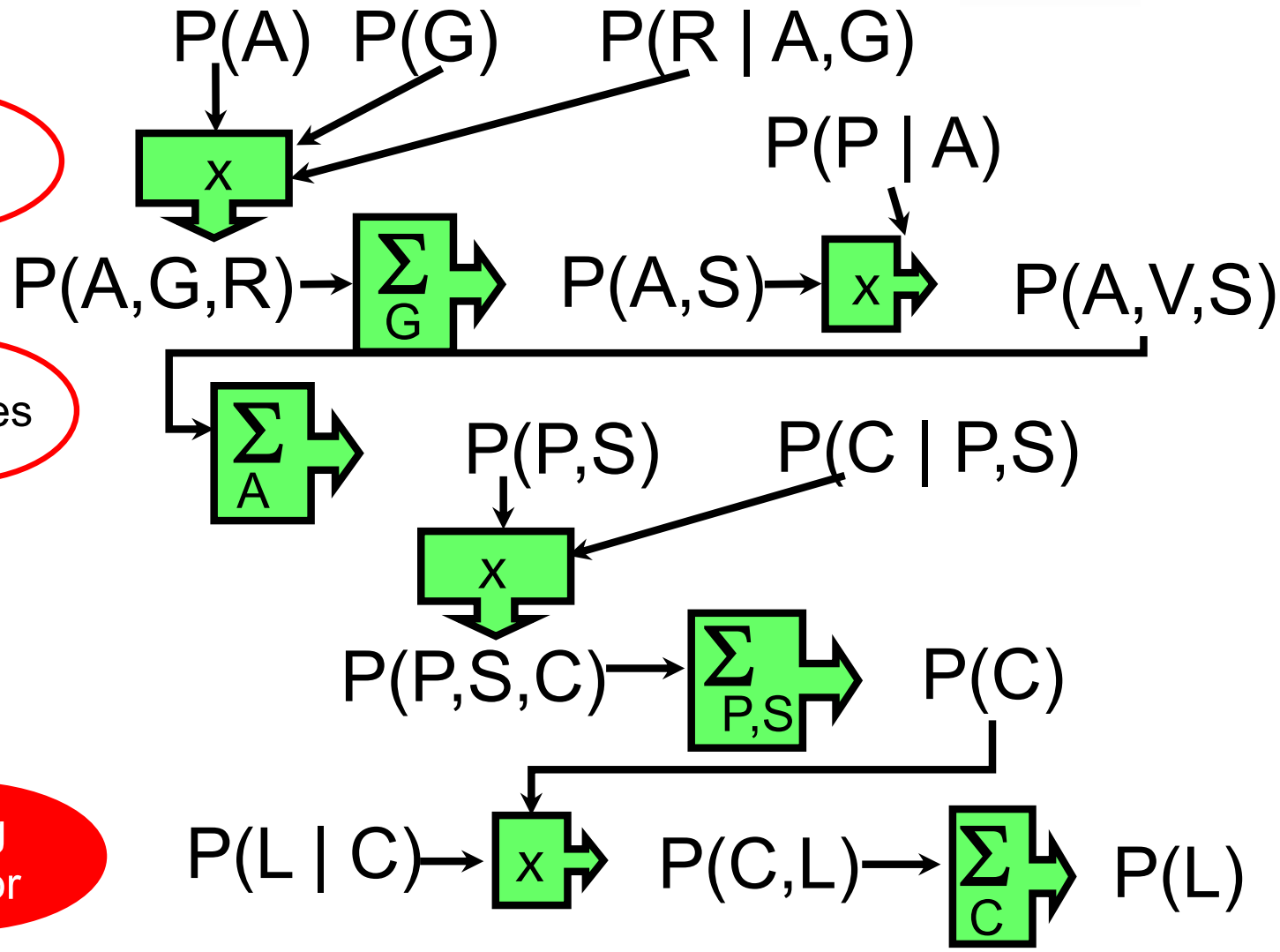
Main observation: Σ and ∏ commute

$\Sigma_a (P_1 \times P_2) = (\Sigma_a P_1) \times P_2$ if A is not in P_2

Complete example: Let's comput P(L)

elimination order is : G,A,V,R,K





Exponentiall in the size of the largest (induced) factor (table) also called treewidth: 2^3 vs 2^7

As an algorithm, this is called: Variable elimination

Given a BN and a query P(X|e) / P(X,e)

Instantiate evidence **e**

Choose an elimination order over the variables, e.g., $X_1, \ldots, X_n$

Initial *factors* $\{f_1,\ldots,f_n\}$: $f_i = P(X_i|\mathbf{Pa}_{X_i})$ (CPT for X_i)

For i = 1 to n, if $X_i \notin \{X,\mathbf{E}\}$

- Collect factors $f_1,\ldots,f_k$ that include X_i
- Generate a new factor by eliminating X_i from these factors

$$g = \sum_{X_i} \prod_{j=1}^{k} f_j$$

- Variable X_i has been eliminated! Add g to the set of factors

Normalize P(X,**e**) to obtain P(X|**e**)

What have we learned so far?

Uncertainty is omnipresent

Uncertainty can be captured using probability distributions

Graphical models are compact encodings of probability distributions

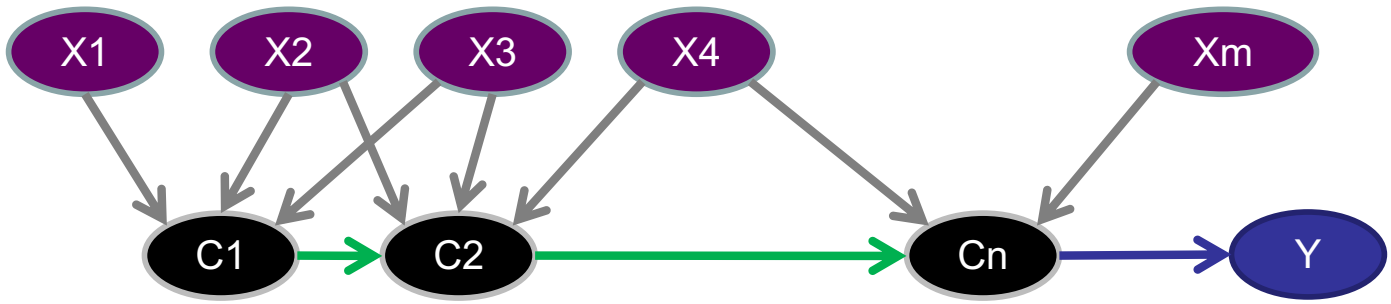
They lead to „efficient“ algorithms for inference such as Variablen-Elimination

Mission Completed? No …

Theorem: Inference (even approximate) in Bayesian networks is NP-hard (#P; via reduction to 3-SAT)

$$\underbrace{(\overline{X}_1 \vee X_2 \vee X_3)}_{C_1} \wedge \underbrace{(\overline{X}_2 \vee X_3 \vee X_4)}_{C_2} \wedge \ldots \underbrace{\quad}_{C_n}$$

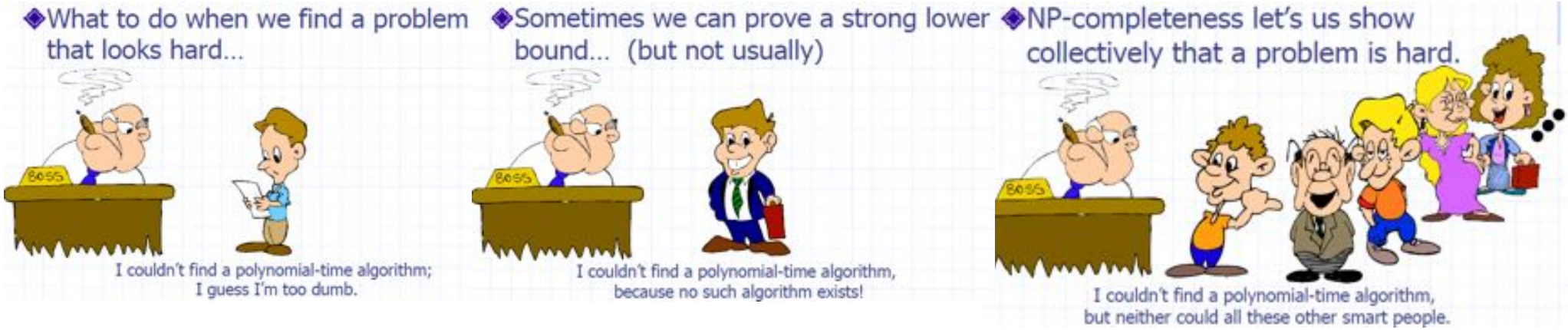
0.5/0.5 prior

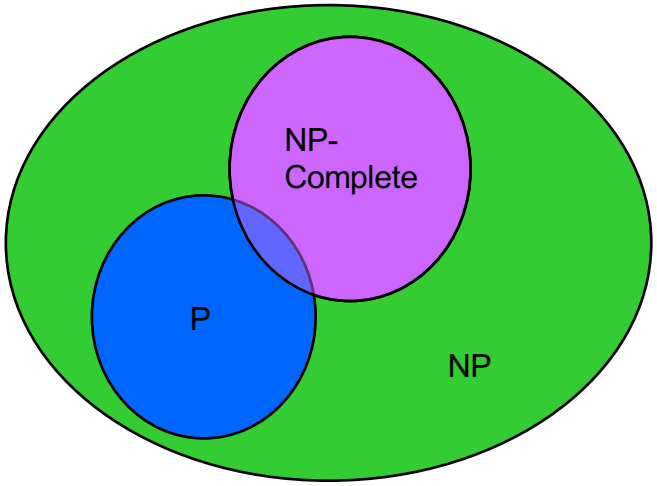


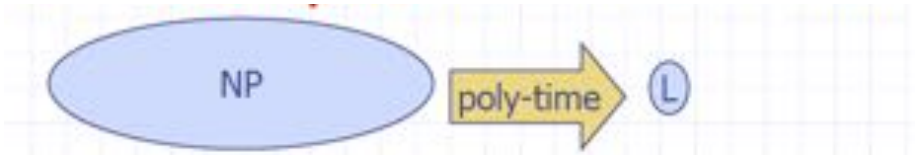
True with prob 1 iff clause 1 is satisfied

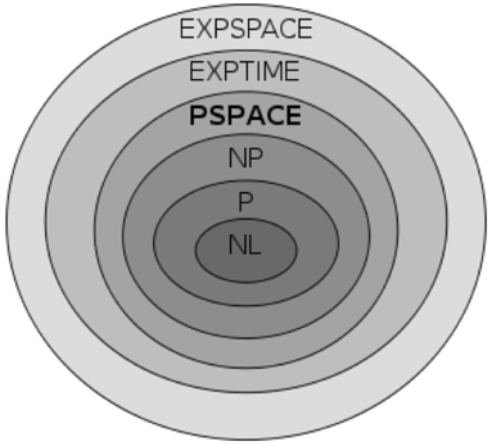
True with prob 1 iff clauses i and i-1 are satisfied

P(Y=t)>0 iff 3-SAT formula is satisfiable


What to do when we find a problem that looks hard...
BOSS
I couldn't find a polynomial-time algorithm; I guess I'm too dumb.
Sometimes we can prove a strong lower bound... (but not usually)
BOSS
I couldn't find a polynomial-time algorithm, because no such algorithm exists!
NP-completeness let's us show collectively that a problem is hard.
BOSS
I couldn't find a polynomial-time algorithm, but neither could all these other smart people.


NP-
Complete
P
NP


NP
poly-time
L


EXPSPACE
EXPTIME
PSPACE
NP
P
NL

What have we learnt about Bayesian networks?

- Bayesian networks (BNs) encode joint distributions
- They are DAGs

  (nodes = RVs, edges = dependencies)
- Inference is NP-hard
- Variable Elimination is one of the most basic inference approaches; there are many other inference approaches
- We have skipped learning BNs

Roadmap of the course

1 Basics of graphical models

2a Exploiting symmetries for inference

2b Sum-product networks

A simple example

Guy van den Broeck
UCLA

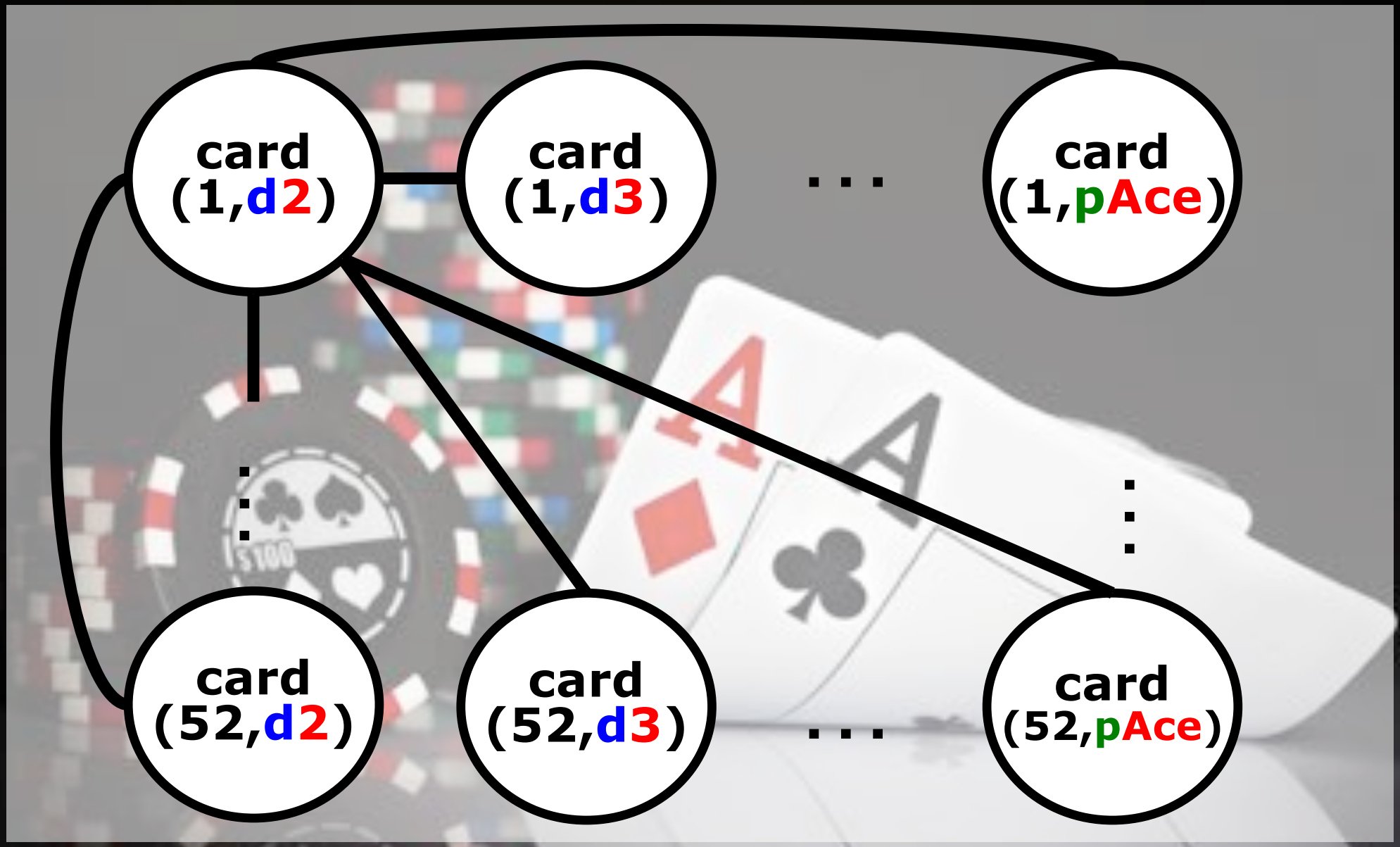
What is the probability that the first card of a randomly shuffled deck with 52 cards is an Ace?

How would a machine solve this?

One option is to treat this as an inference problem within in a graphical model

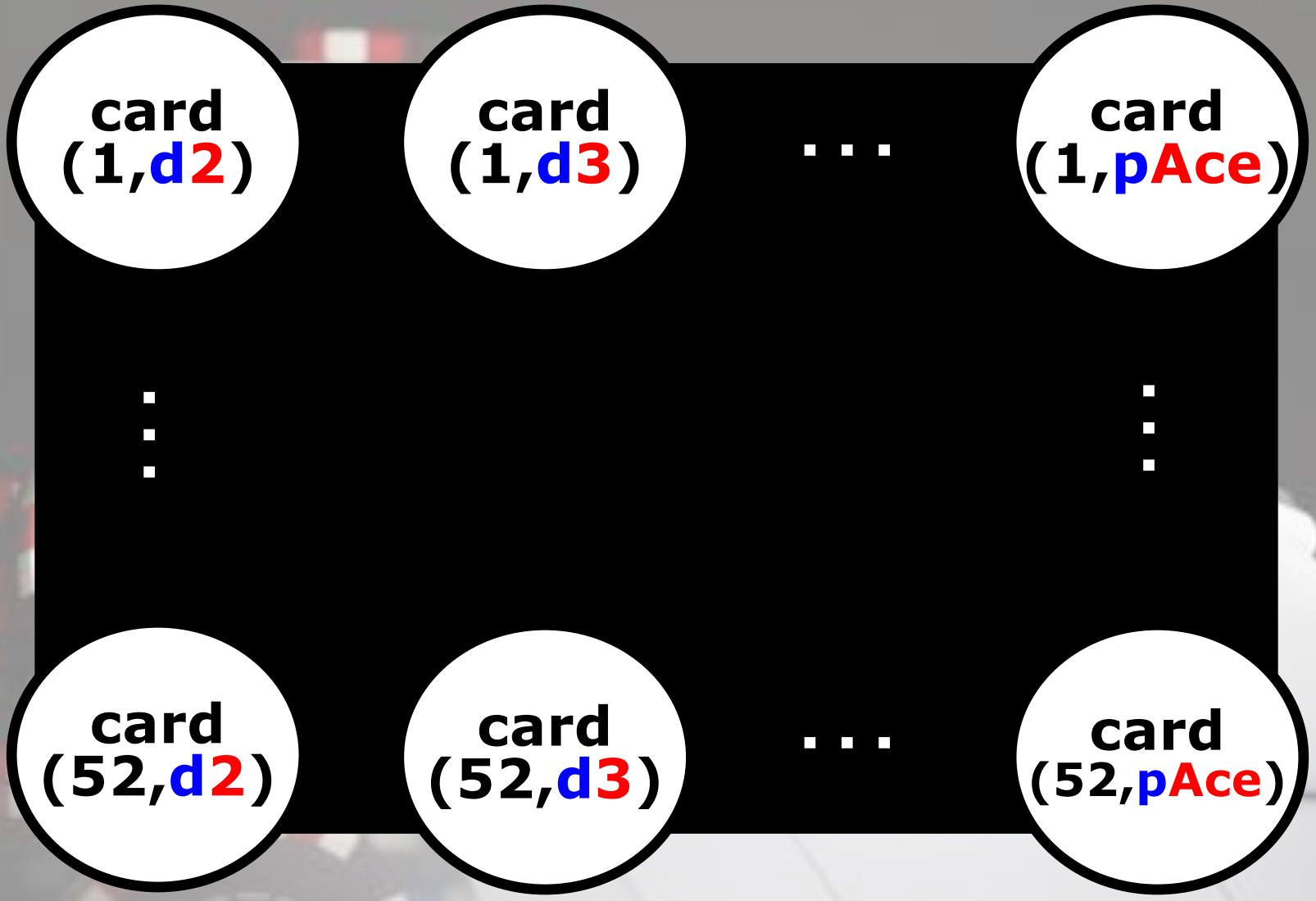
A simple example

Guy van den Broeck
UCLA



A simple example

Guy van den Broeck
UCLA



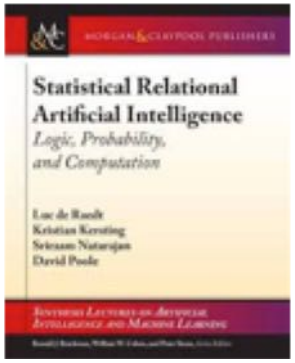

We do not want to write down all the rules!

Faster modelling

Let's use programming abstractions such as e.g.

w1:∀p,x,y: card(P,X),card(P,Y)⇒x=y
w2:∀c,x,y: card(X,C),card(Y,C)⇒x=y

We do not want to write down all the rules!

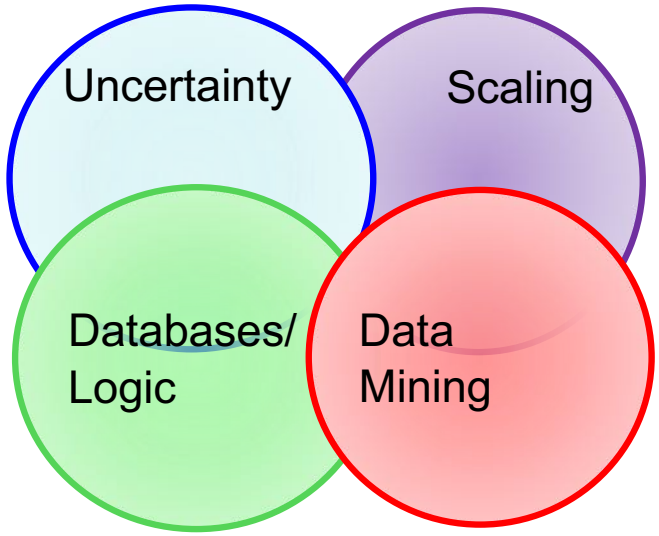


De Raedt, Kersting, Natarajan, Poole, Statistical Relational Artificial Intelligence: Logic, Probability, and Computation. Morgan and Claypool Publishers, ISBN: 9781627058414, 2016.

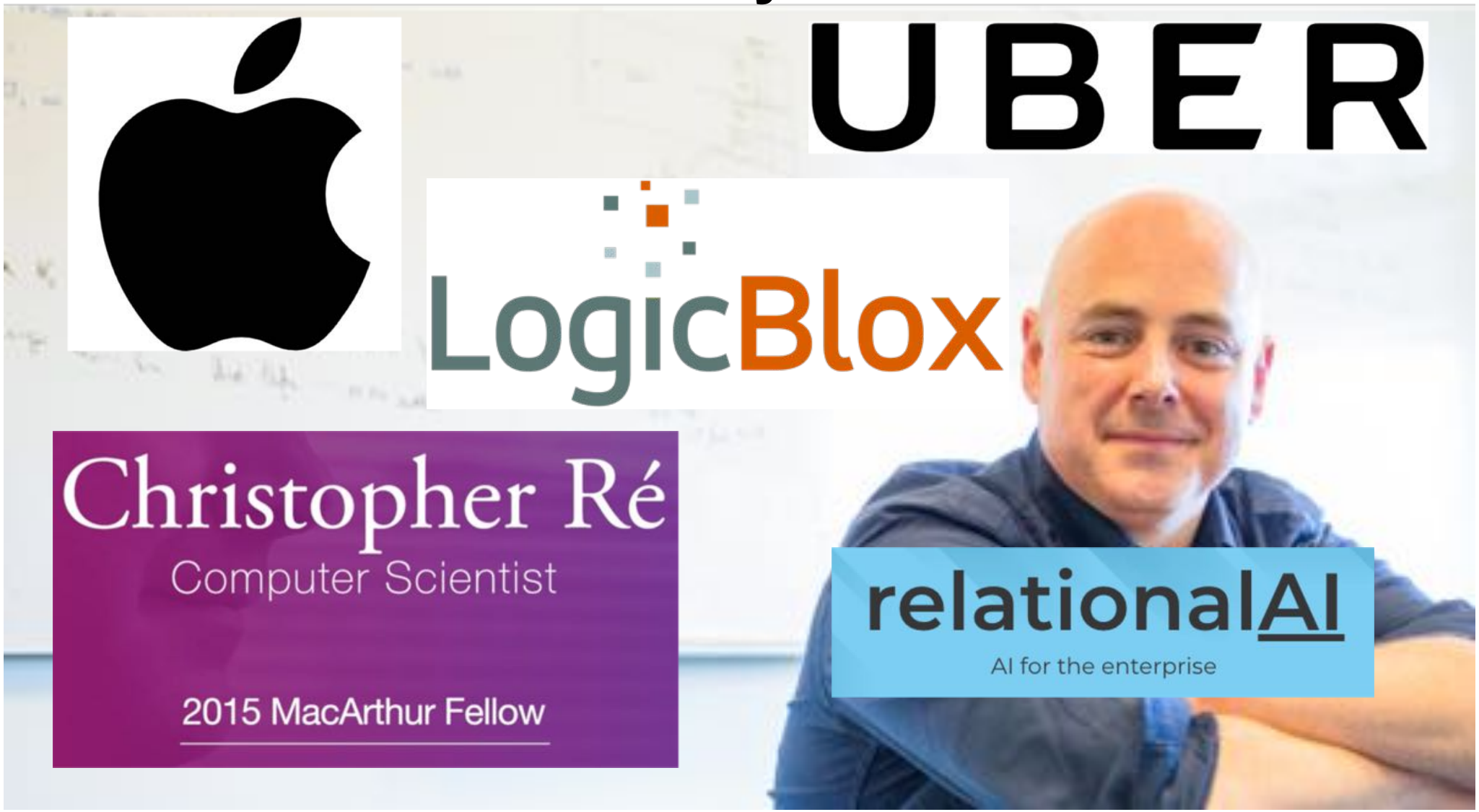
Crossover of ML/AI with data & programming abstractions

make the ML/AI expert more effective

increases the number of people who can successfully build ML/AI applications



Not convinced? LogicBlox, RelationalAI, Apple and Uber invest(ed) hundreds of millions of dollars into this AI/ML Systems view



A simple example

Guy van den Broeck
UCLA

card (1,d2)

card (1,d3)

...

card (1,pAce)

What about inference?

card (52,d2)

card (52,d3)

...

card (52,pAce)

A simple example

Guy van den Broeck
UCLA

card (1,d2)

card (1,d3)

...

card (1,pAce)

⋮

No independencies.
Fully connected.

2^{2704} states

⋮

card (52,d2)

card (52,d3)

...

card (52,pAce)

A simple example

Guy van den Broeck
UCLA

card (1,d2)

card (1,d3)

...

card (1,pAce)

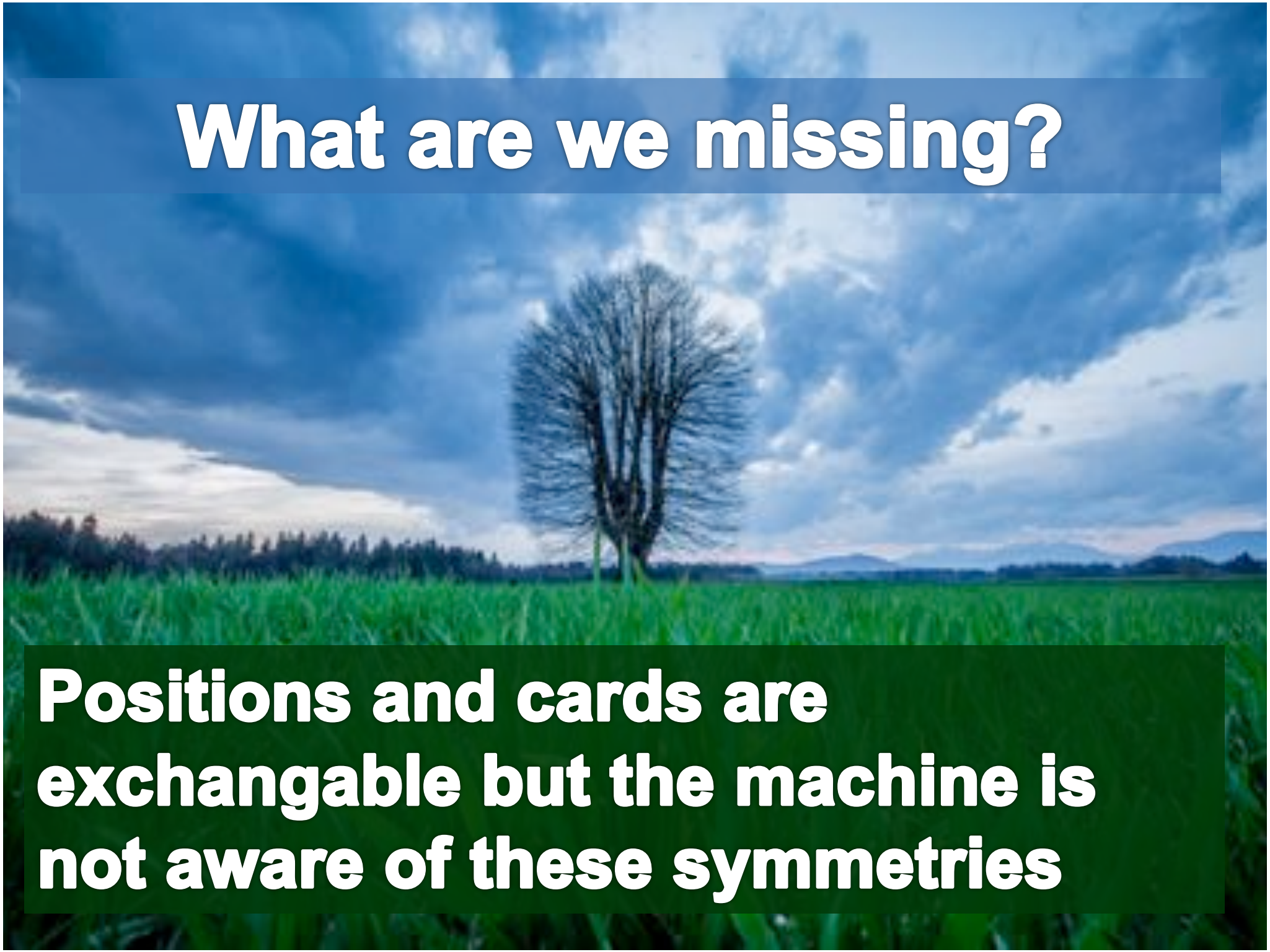
⋮ **A machine will not solve the problem** ⋮

card (52,d2)

card (52,d3)

...

card (52,pAce)



What are we missing?

Positions and cards are exchangable but the machine is not aware of these symmetries

Faster modelling

Let's use programming abstractions together with symmetry- and language-aware solvers

Faster solvers

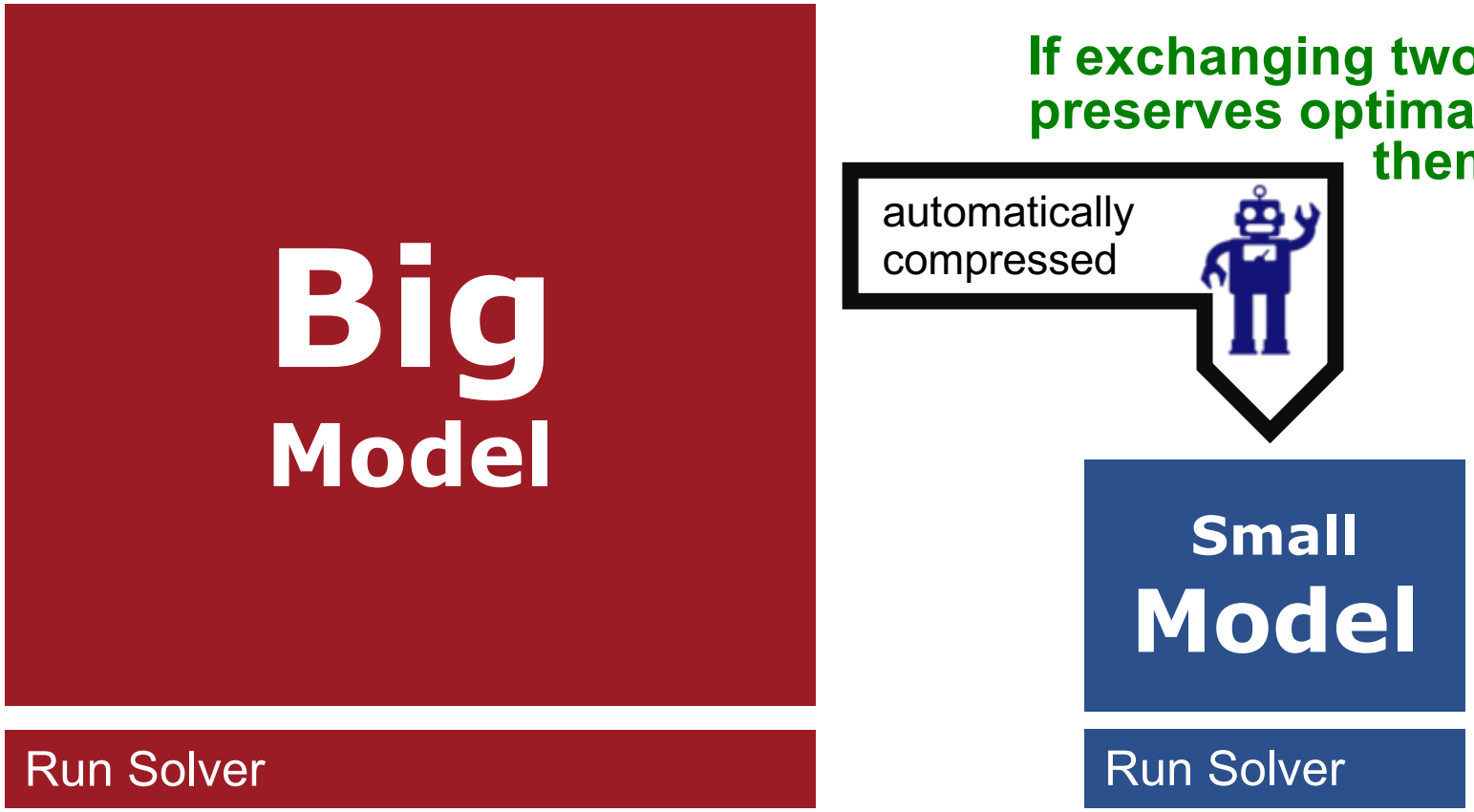
Positions and cards are exchangable but the machine is not aware of these symmetries

Like „-O“ flags known from compilers:

Let the AI machine figure out computational symmetries

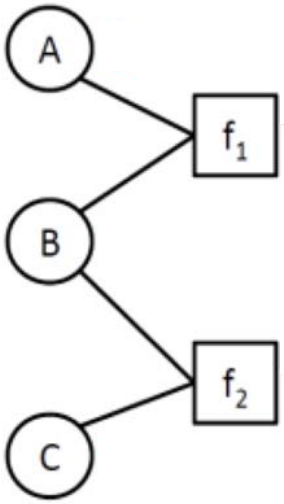
[Mladenov, Ahmadi, Kersting AISTATS´12, Grohe, Kersting, Mladenov, Selman ESA´14, Kersting, Mladenov, Tokmatov AIJ´17]

If exchanging two variables preserves optimality, group them together



Compression: Coloring the graph

[Kersting, Ahmadi, Natarajan UAI'09; Ahmadi, Kersting, Mladenov, Natarajan MLJ'13, Mladenov, Ahmadi, Kersting AISTATS´12, Grohe, Kersting, Mladenov, Selman ESA´14, Kersting, Mladenov, Tokmatov AIJ´17]

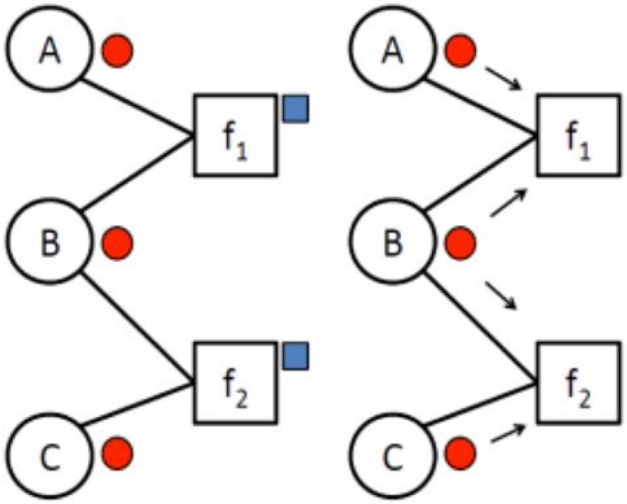


Color nodes initially with the same color, say red

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**

Compression: Pass colors around

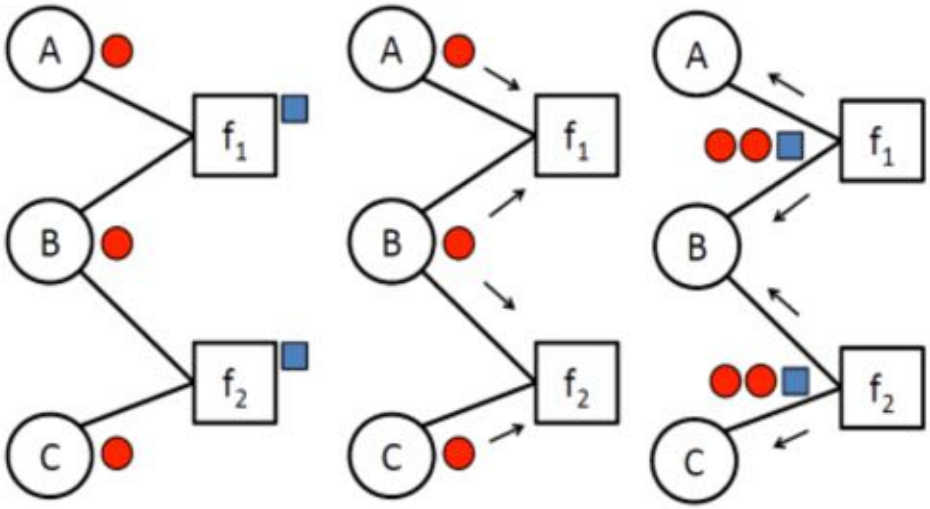
[Kersting, Ahmadi, Natarajan UAI'09; Ahmadi, Kersting, Mladenov, Natarajan MLJ'13, Mladenov, Ahmadi, Kersting AISTATS´12, Grohe, Kersting, Mladenov, Selman ESA´14, Kersting, Mladenov, Tokmatov AIJ´17]



1. Each factor collects the colors of its neighboring nodes

Compression: Pass colors around

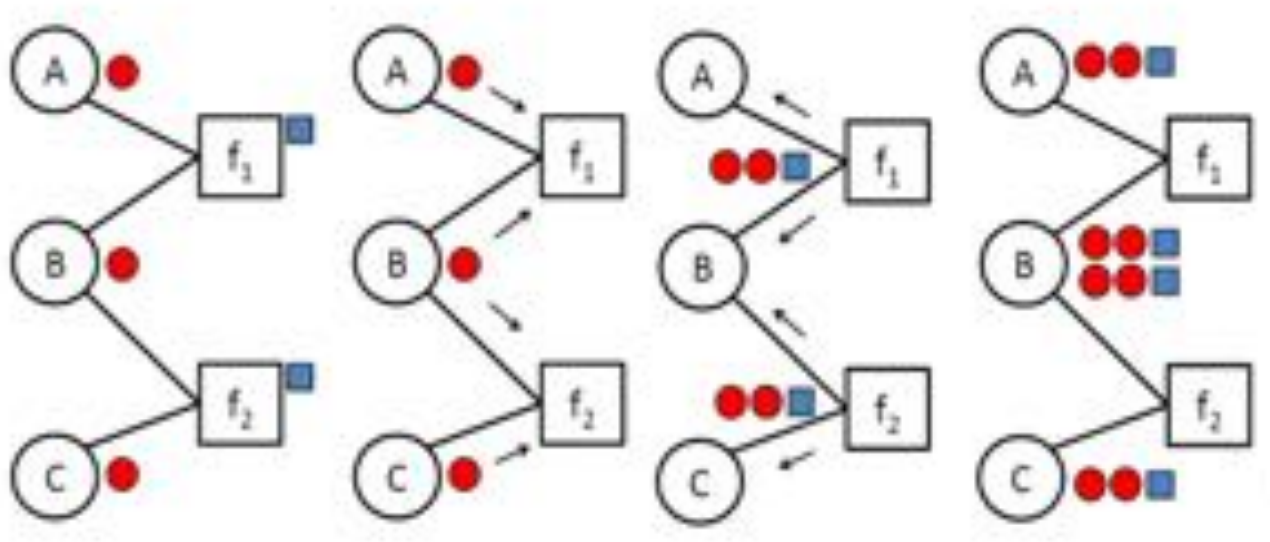
[Kersting, Ahmadi, Natarajan UAI'09; Ahmadi, Kersting, Mladenov, Natarajan MLJ'13, Mladenov, Ahmadi, Kersting AISTATS´12, Grohe, Kersting, Mladenov, Selman ESA´14, Kersting, Mladenov, Tokmatov AIJ´17]



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

Compression: Pass colors around

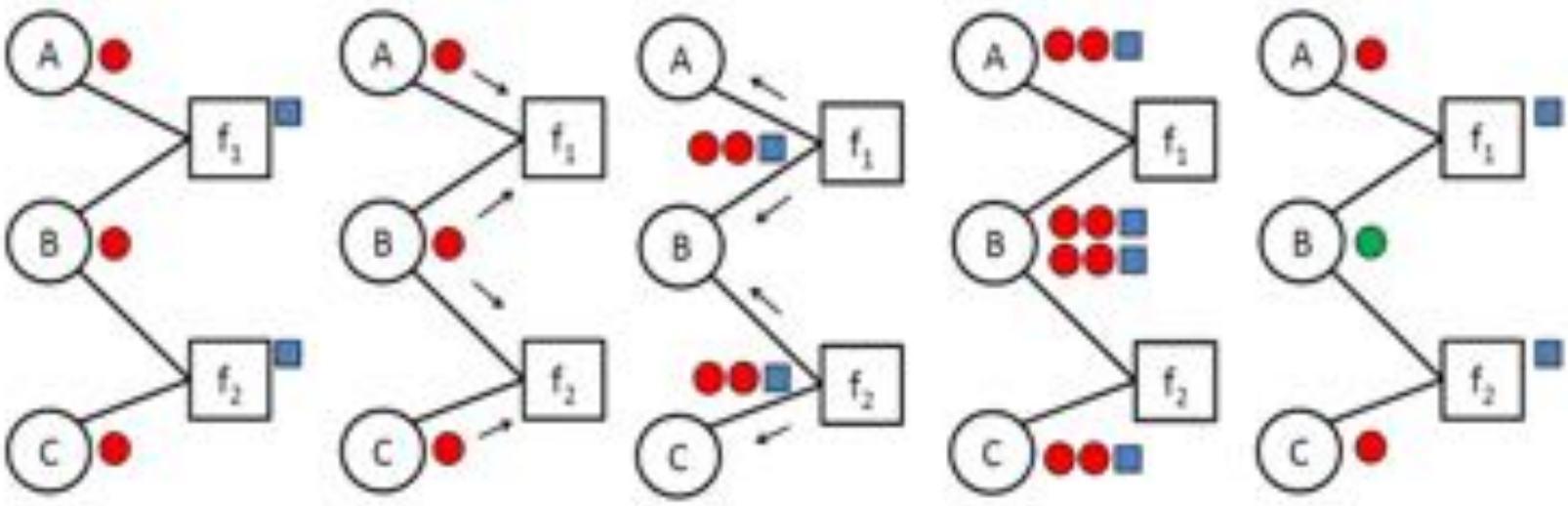
[Kersting, Ahmadi, Natarajan UAI'09; Ahmadi, Kersting, Mladenov, Natarajan MLJ'13, Mladenov, Ahmadi, Kersting AISTATS´12, Grohe, Kersting, Mladenov, Selman ESA´14, Kersting, Mladenov, Tokmatov AIJ´17]



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

Compression: Pass colors around

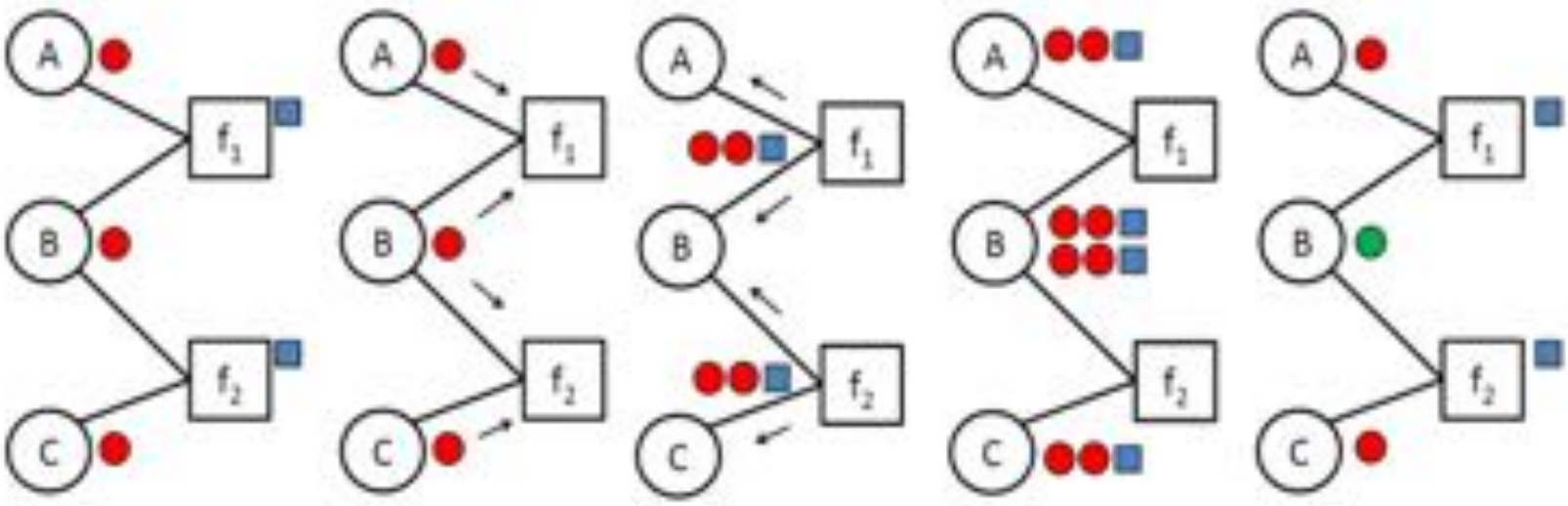
[Kersting, Ahmadi, Natarajan UAI'09; Ahmadi, Kersting, Mladenov, Natarajan MLJ'13, Mladenov, Ahmadi, Kersting AISTATS´12, Grohe, Kersting, Mladenov, Selman ESA´14, Kersting, Mladenov, Tokmatov AIJ´17]



1. Each factor collects the colors of its neighboring nodes
2. Each factor „signs“ its 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 colors around

[Kersting, Ahmadi, Natarajan UAI'09; Ahmadi, Kersting, Mladenov, Natarajan MLJ'13, Mladenov, Ahmadi, Kersting AISTATS´12, Grohe, Kersting, Mladenov, Selman ESA´14, Kersting, Mladenov, Tokmatov AIJ´17]



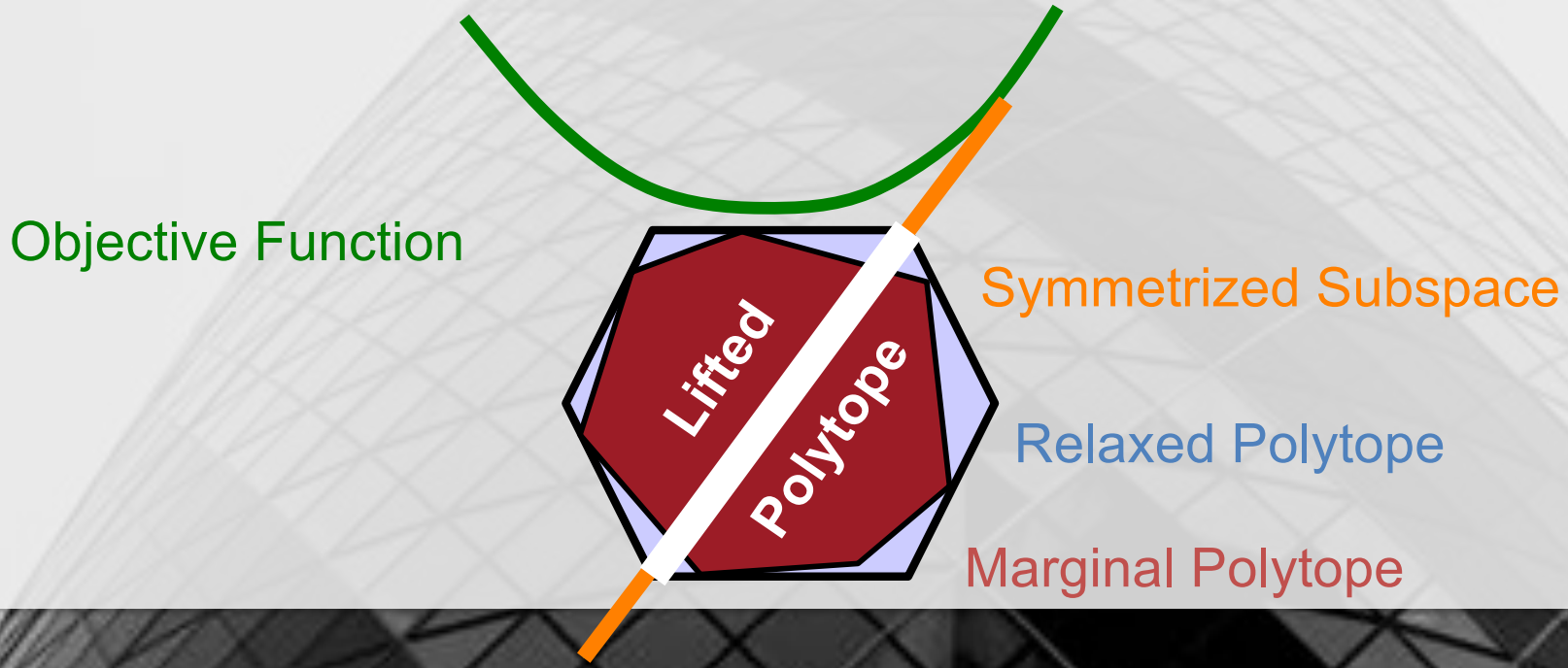
1. Each factor collects the colors of its neighboring nodes
2. Each factor „signs“ its color signature with its own color
3. Each node collects the signatures of its neighboring factors
4. Nodes are recolored according to the collected signatures
5. If no new color is created stop, otherwise go back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

Why does this work?

[Mladenov, Globerson, Kersting UAI ´14, AISTATS ´14, Mladenov, Kersting UAI ´15]

Approximate probabilistic inference closely connected to LPs

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

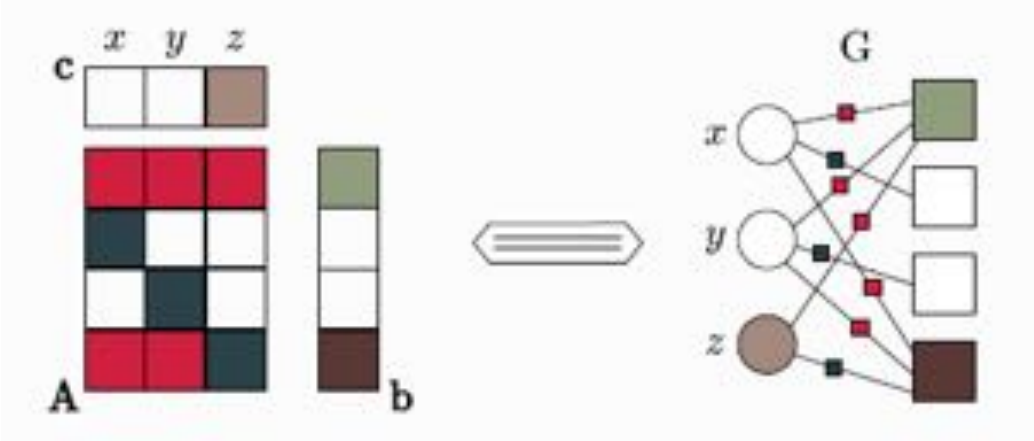


Why does this work?

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

TECHNISCHE UNIVERSITÄT DARMSTADT

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 **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; they are averagers. 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

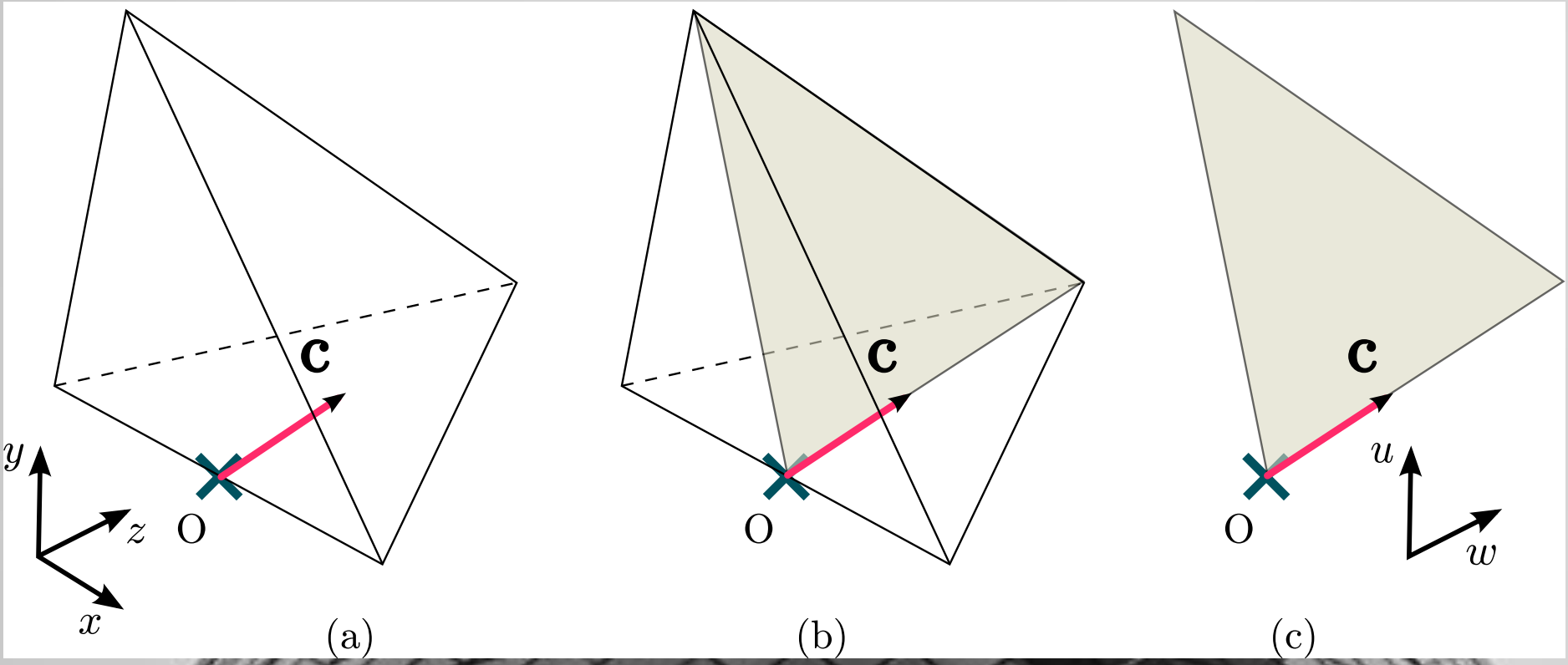
$$\mathbf{X}_P = \mathbf{B}\mathbf{B}^T$$

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

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

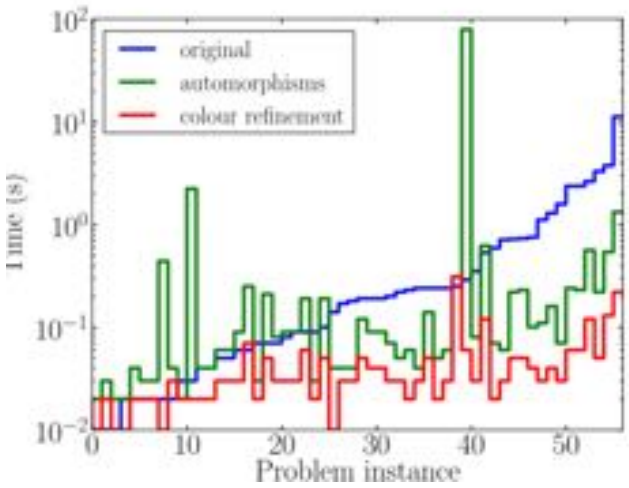


Feasible region of LP and the objective vectors

Span of the fractional auto-morpishm of the LP

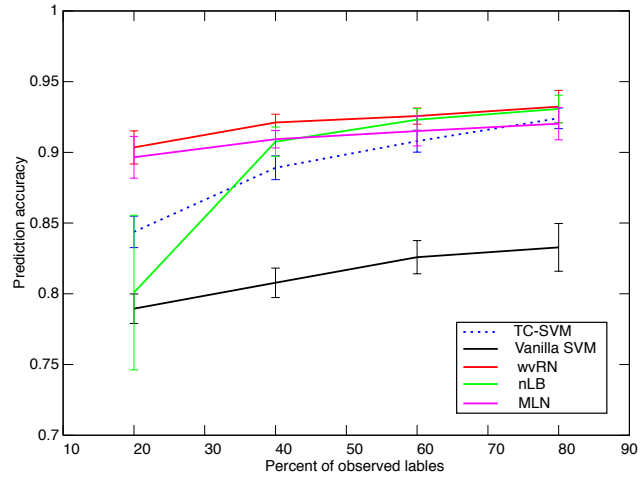
Projections of the feasible region onto the span of the fractional auto-morphism

WL induces a Fractional Automorphism of the LP



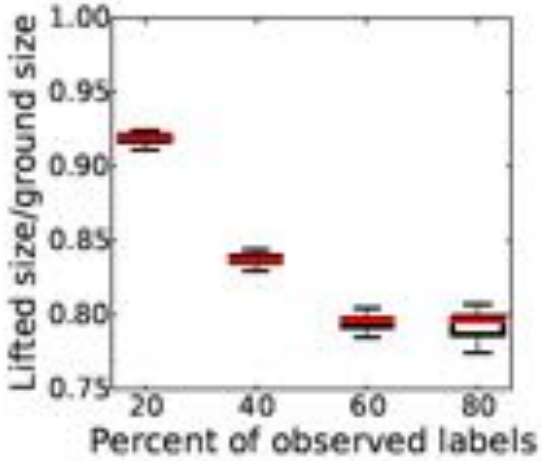
Margout‘s ILPs with symmetries (relaxed)

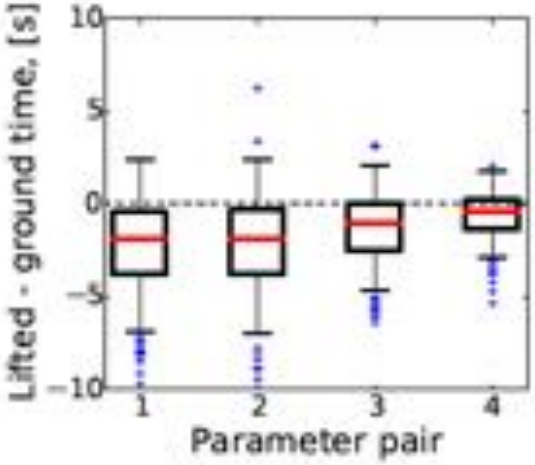
TECHNISCHE UNIVERSITÄT DARMSTADT

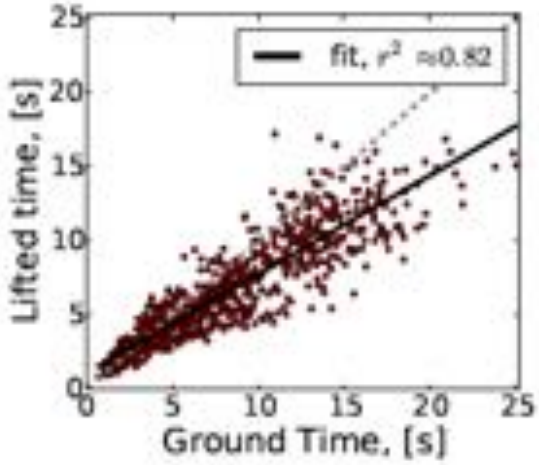


Collective Classification

Cora (most common vs. rest)





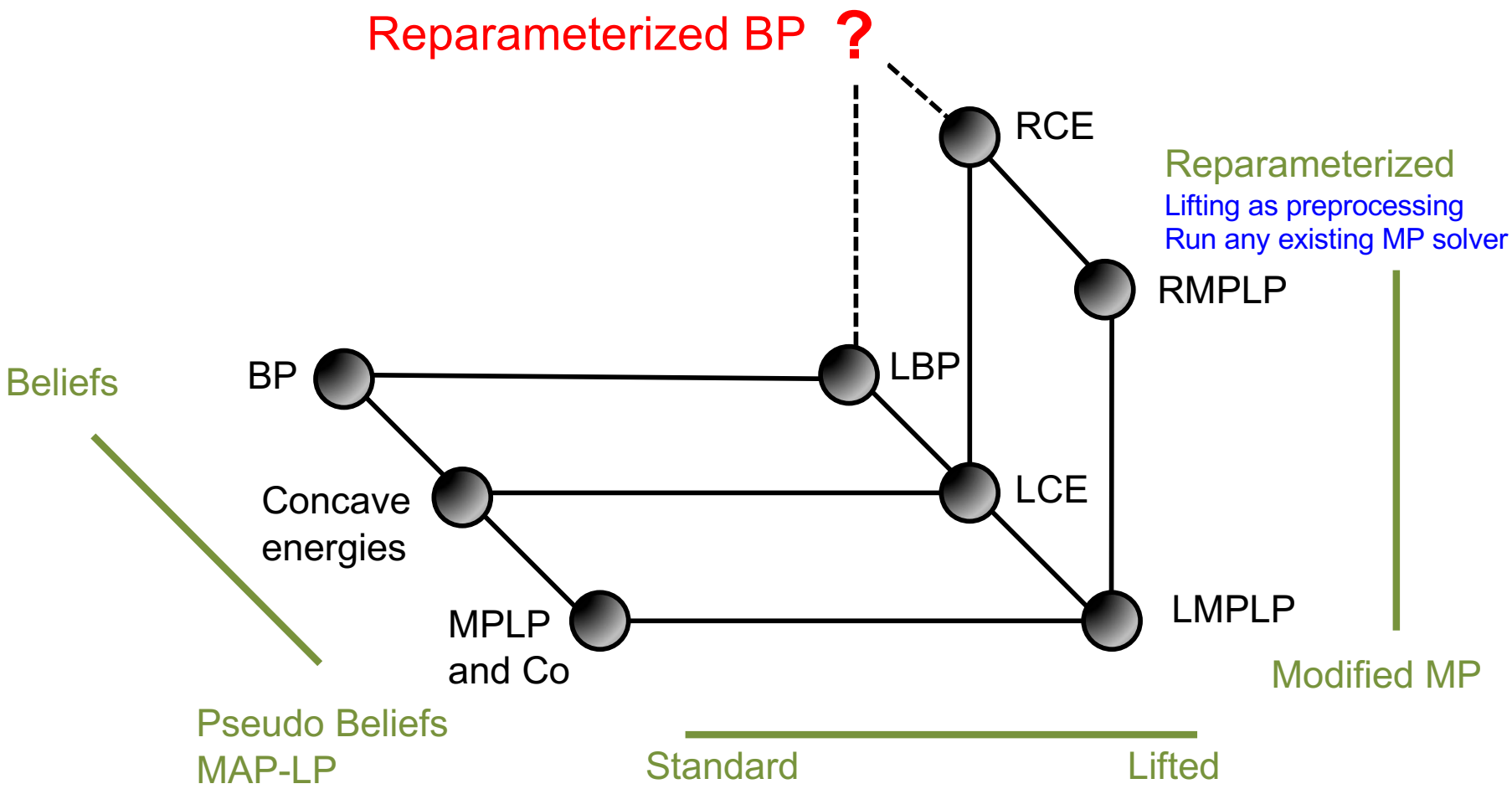


The more observed the more lifting

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

Lifted probabilistic inference = Inference in a smaller, reparameterized model

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



[Mladenov, Kleinhans, Kersting AAAI 2017]

BTW, this also works for Convex Quadratic Programs

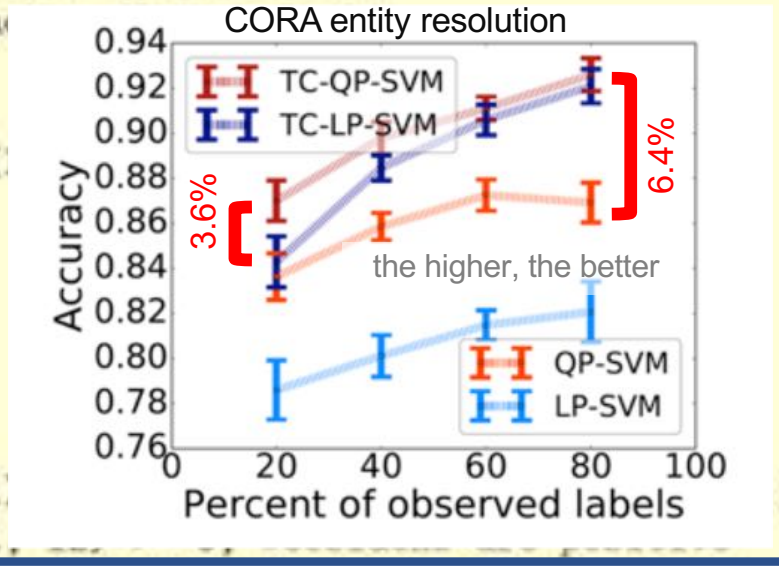
$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}\in\mathcal{D}} J(\boldsymbol{x})$$
$$J(\boldsymbol{x}) = \boldsymbol{x}^T Q \boldsymbol{x} + \boldsymbol{c}^T \boldsymbol{x}$$
$$\mathcal{D} = \{\boldsymbol{x} : A\boldsymbol{x} \leqslant \boldsymbol{b}\}$$

(1) A relational language for QPs

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

#labeled examples should be on the correct side
subject

#slack
subject

#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,
```

Citing papers should be on the same side of the hyperplane

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

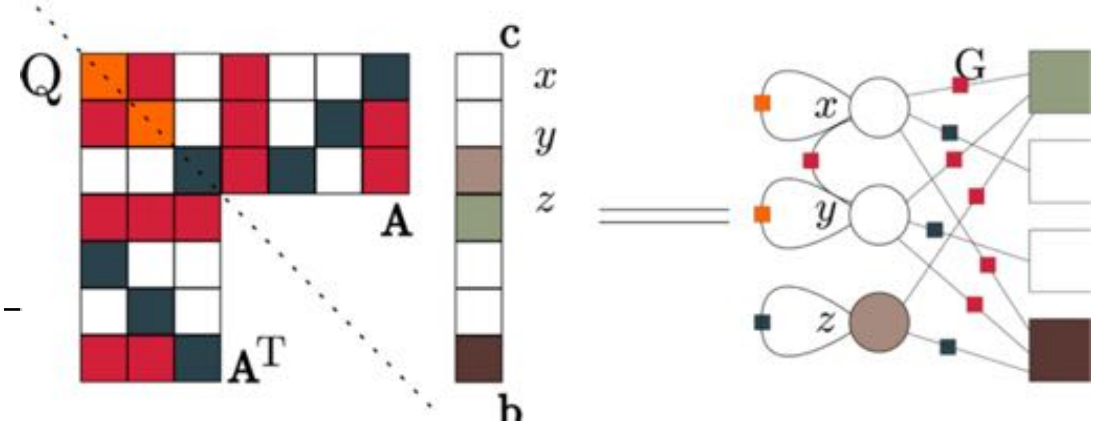


(2) Lifted inference for convex QPs

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

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

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



Industrial Strength Solvers such as CPLEX and GUROBI are deploying this

What have we learnt about lifted inference?

- Learning (rich) representations is a central problem of machine learning
- (Fractional) symmetry / group theory provide a natural foundation for learning representations
- Symmetries = “unimportant” variants of data (graphs, relational structures, …)
- “Unimportant” variants get grouped together

However, inference and modelling are still “hard”

For Systems AI we have to provide a set of tools for understanding data that require minimal expert input

The Automatic Statistician

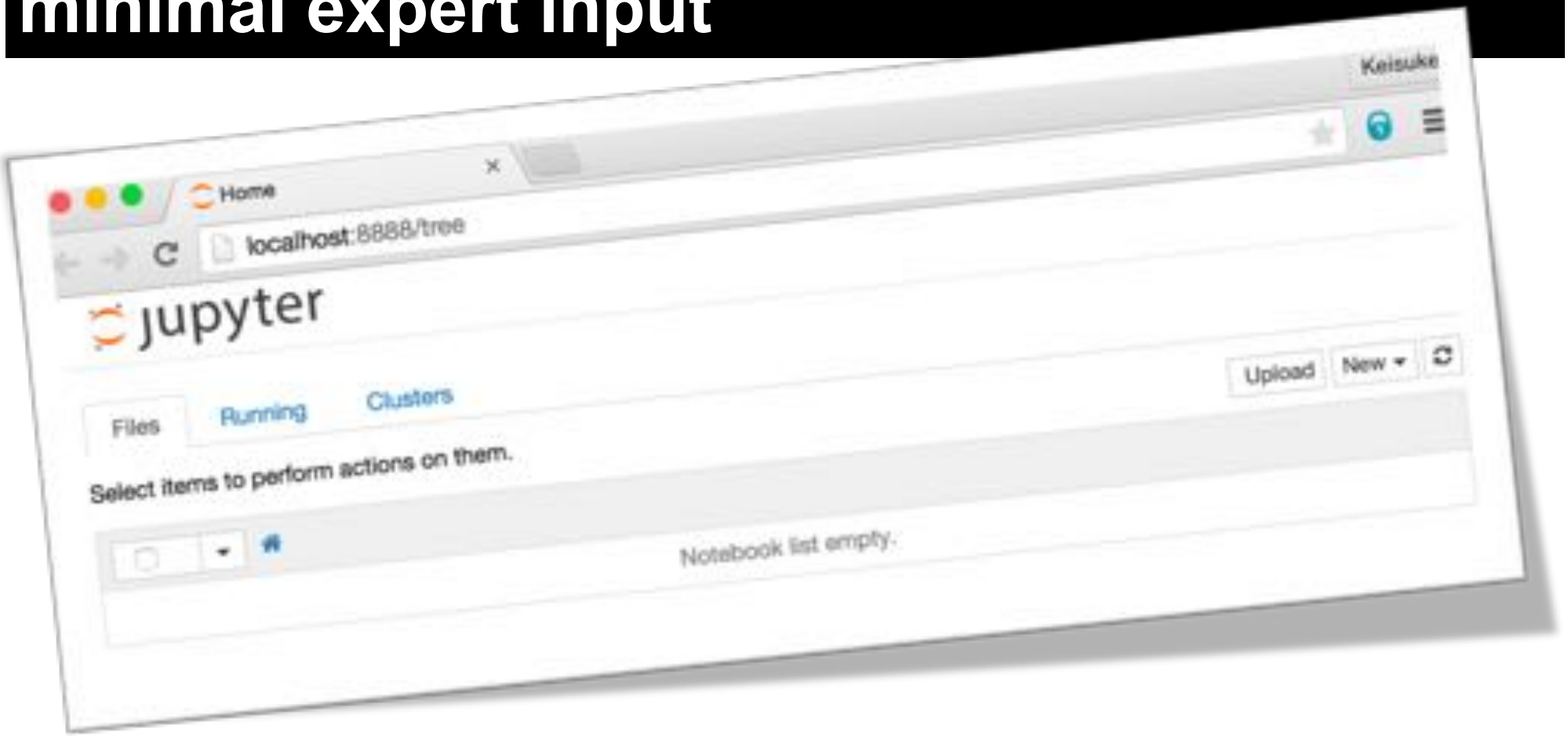
A system which explores an open-ended space of statistical models to discover a good explanation of the data, and then produces a detailed report with figures and natural-language text

Only regression so far!

Llyod, Duvenaud, Ghahramani
U. Cambridge

Grosse, Tenenbaum
MIT

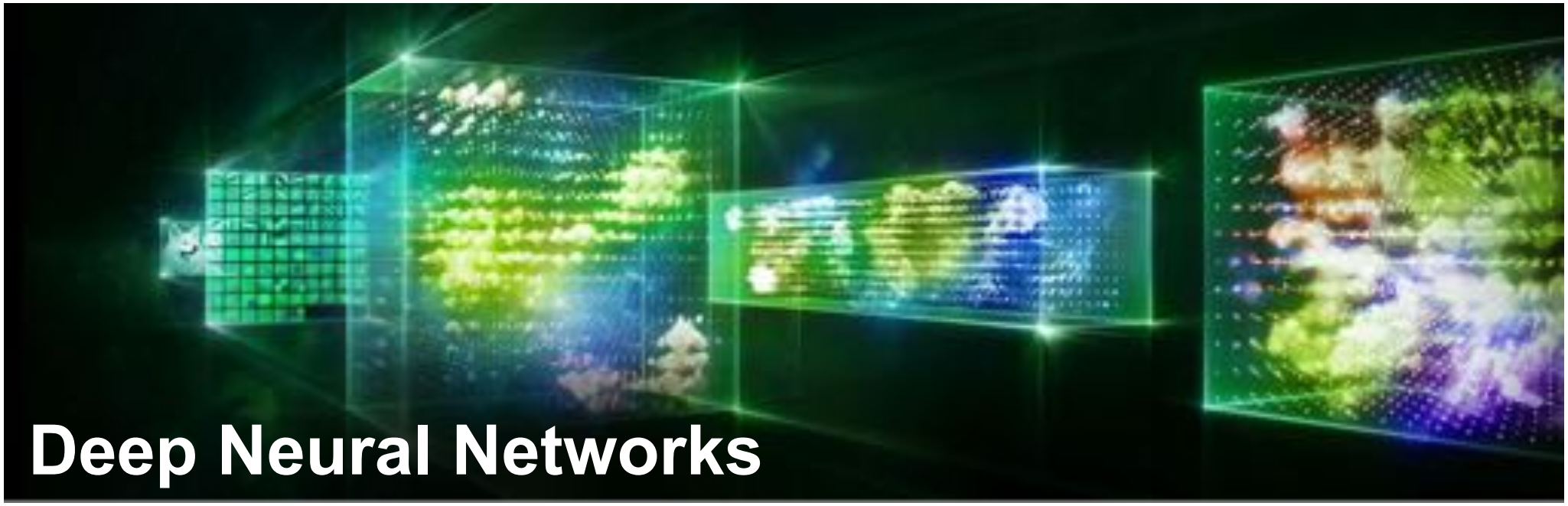
For Systems AI we have to provide a set of tools for understanding data that require minimal expert input



Instead of starting with an empty notebook …

For Systems AI we have to provide a set of tools for understanding data that require minimal expert input

the machine automatically compiles one for you!

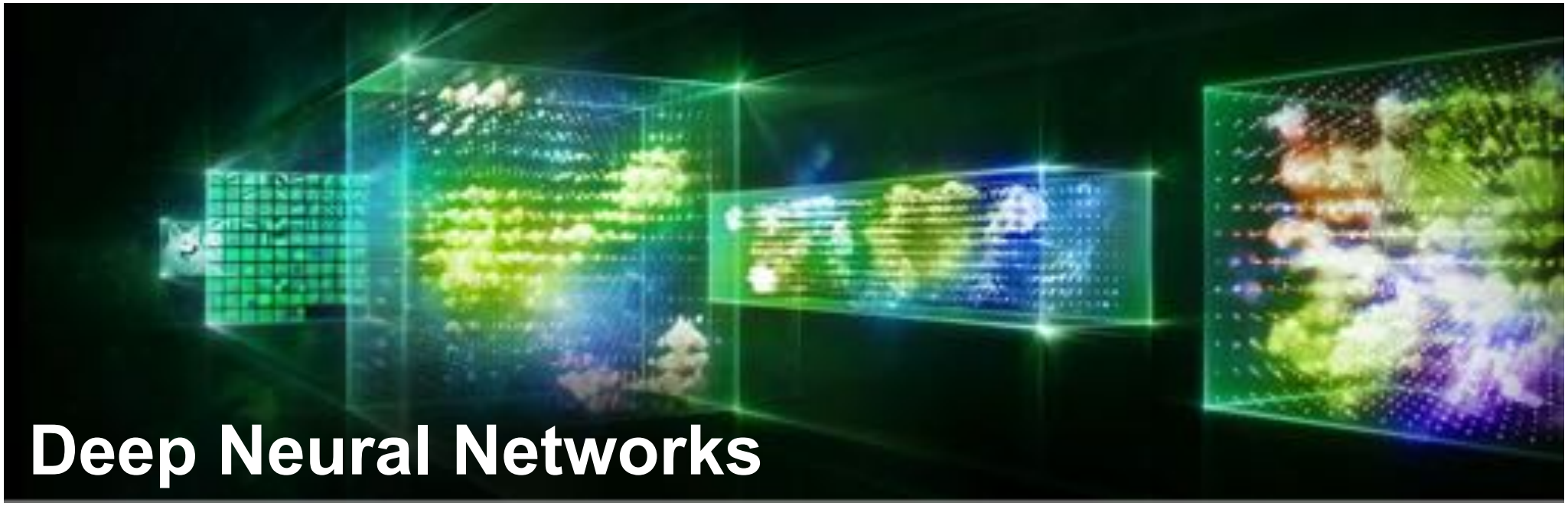


Deep Neural Networks

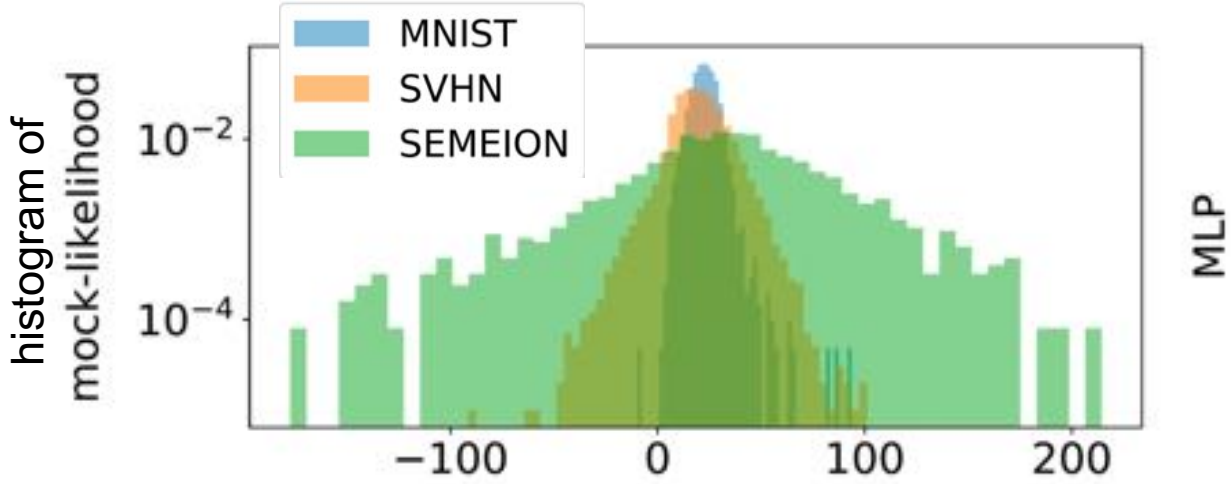
Potentially much more powerful than shallow architectures, represent computations [Bengio, 2009]

But …

- **Often no probabilistic semantics**
- **Learning requires extensive efforts**

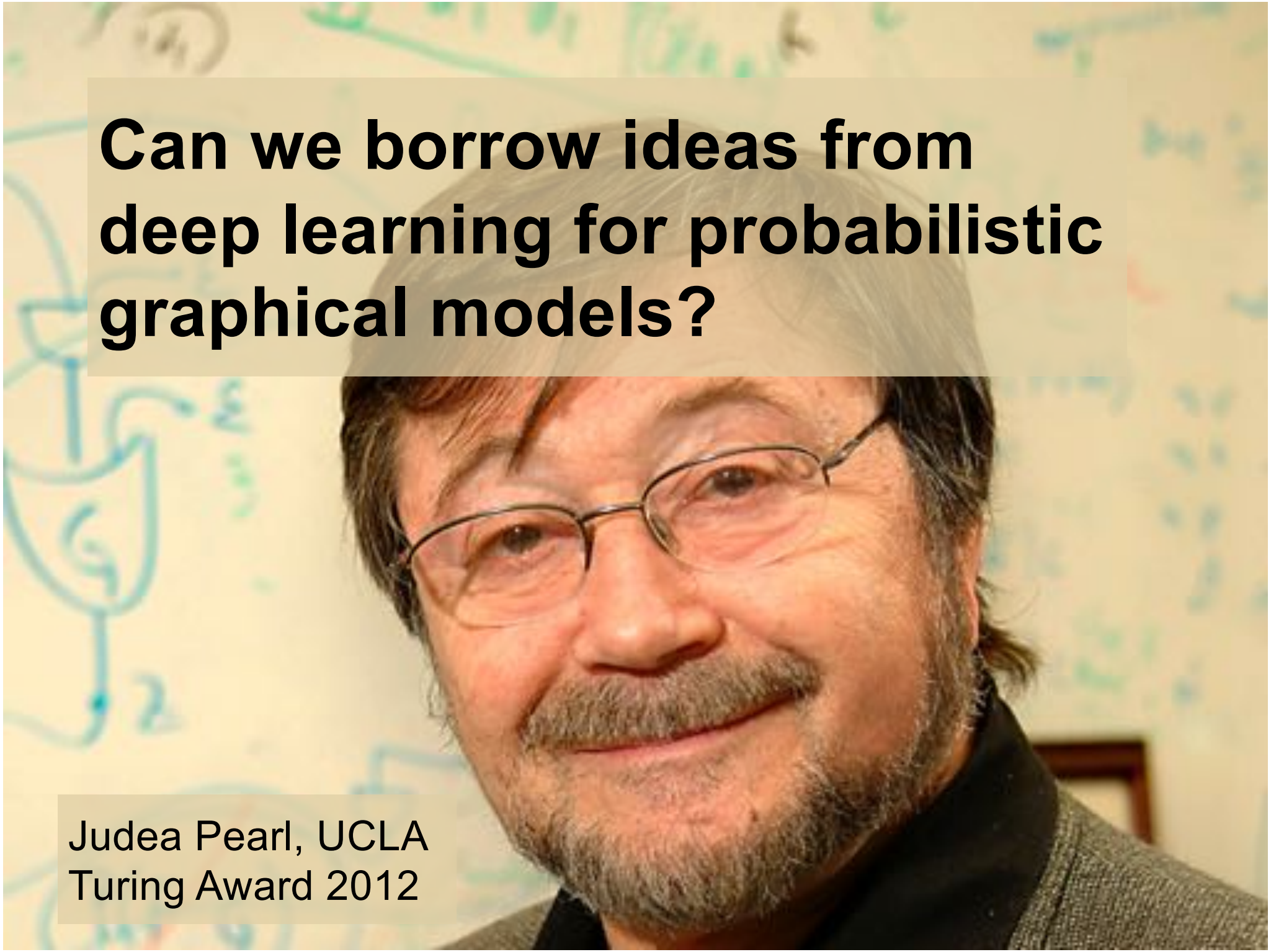


Deep Neural Networks



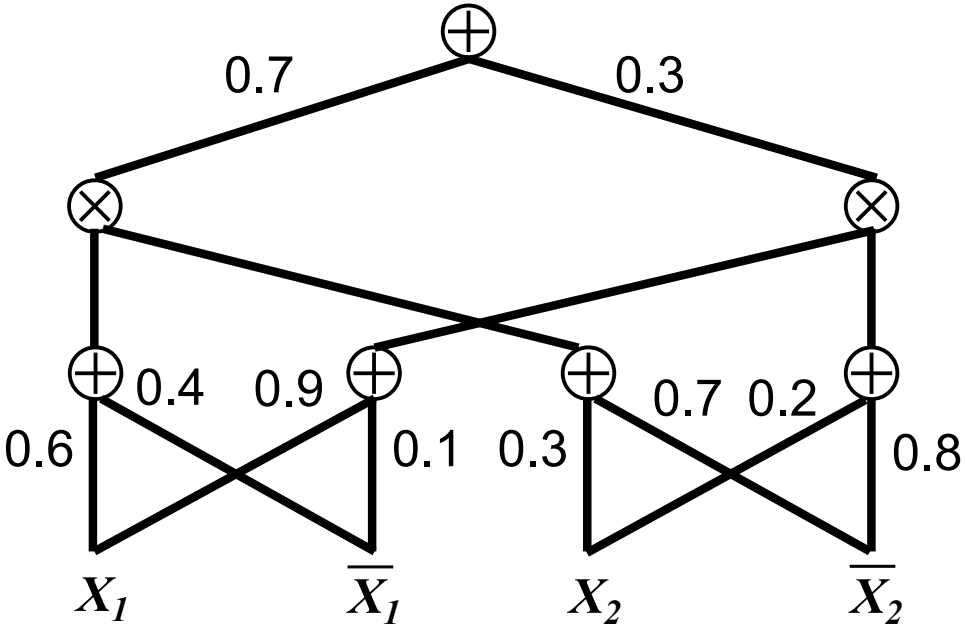
Deep neural networks may not be faithful probabilistic models

Can we borrow ideas from deep learning for probabilistic graphical models?



Judea Pearl, UCLA
Turing Award 2012

Deep Probabilistic Modelling using Sum-Product Networks



Computational graph (kind of TensorFlow graphs) that encodes how to compute probabilities

Inference is Linear in Size of Network

Alternative Representation: Graphical Models as Deep Networks

X_1	X_2	$P(X)$
1	1	0.4
1	0	0.2
0	1	0.1
0	0	0.3

$$
\begin{aligned}
P(X) = {} & 0.4 \cdot \mathrm{I}[X_1{=}1] \cdot \mathrm{I}[X_2{=}1] \\
& + 0.2 \cdot \mathrm{I}[X_1{=}1] \cdot \mathrm{I}[X_2{=}0] \\
& + 0.1 \cdot \mathrm{I}[X_1{=}0] \cdot \mathrm{I}[X_2{=}1] \\
& + 0.3 \cdot \mathrm{I}[X_1{=}0] \cdot \mathrm{I}[X_2{=}0]
\end{aligned}
$$

Alternative Representation: Graphical Models as Deep Networks

X_1	X_2	$P(X)$
1	**1**	**0.4**
1	0	0.2
0	1	0.1
0	0	0.3

$$
\begin{aligned}
P(X) = \; & \mathbf{0.4 \cdot I[X_1{=}1] \cdot I[X_2{=}1]} \\
& + 0.2 \cdot \mathrm{I}[X_1{=}1] \cdot \mathrm{I}[X_2{=}0] \\
& + 0.1 \cdot \mathrm{I}[X_1{=}0] \cdot \mathrm{I}[X_2{=}1] \\
& + 0.3 \cdot \mathrm{I}[X_1{=}0] \cdot \mathrm{I}[X_2{=}0]
\end{aligned}
$$

Shorthand for Indicators

X_1	X_2	$P(X)$
1	1	0.4
1	0	0.2
0	1	0.1
0	0	0.3

$$
\begin{aligned}
P(X) = 0.4 \cdot X_1 \cdot X_2 \\
+ 0.2 \cdot X_1 \cdot \overline{X_2} \\
+ 0.1 \cdot \overline{X_1} \cdot X_2 \\
+ 0.3 \cdot \overline{X_1} \cdot \overline{X_2}
\end{aligned}
$$

Sum Out Variables

X_1	X_2	$P(X)$
1	**1**	**0.4**
1	**0**	**0.2**
0	1	0.1
0	0	0.3

$$\boxed{e\colon X_1 = 1}$$

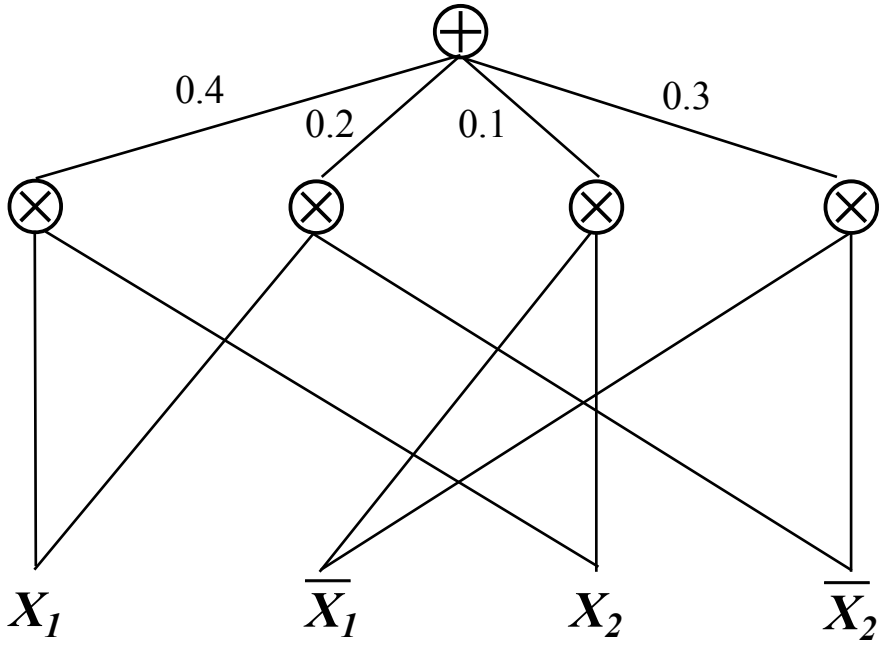
$$\begin{aligned} P(e) = {} & \mathbf{0.4 \cdot X_1 \cdot X_2} \\ & \mathbf{+\, 0.2 \cdot X_1 \cdot \overline{X}_2} \\ & +\, 0.1 \cdot \overline{X}_1 \cdot X_2 \\ & +\, 0.3 \cdot \overline{X}_1 \cdot \overline{X}_2 \end{aligned}$$

Set $X_1 = 1, \overline{X}_1 = 0, \boxed{X_2 = 1, \overline{X}_2 = 1}$

Easy: Set both indicators of X2 to 1

Idea: Deeper Network Representation of a Graphical Model that encodes how to compute probabilities

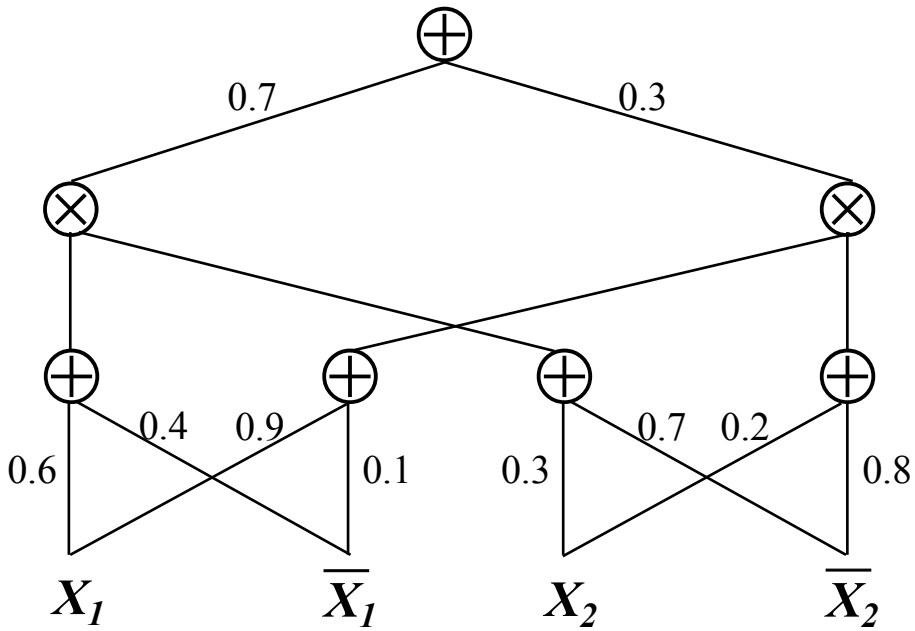
X_1	X_2	$P(X)$
1	1	0.4
1	0	0.2
0	1	0.1
0	0	0.3



Sum-Product Networks* (SPNs)

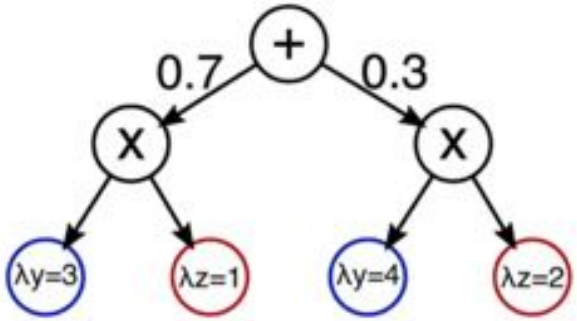
[Poon, Domingos UAI 2011]

A SPN S is a rooted DAG where:
Nodes: Sum, product, input indicator
Weights on edges from sum to children



*SPNs are an instance of Arithmetic Circuits (ACs). ACs have been introduced into the AI literature more than15 years ago as a tractable representation of probability distributions
[Darwiche CACM 48(4):608-647 2001]

Deep Probabilistic Inference Units



SPNs can be compiled into flat, library-free code suitable for embedding in real-time applications and devices

TABLE II

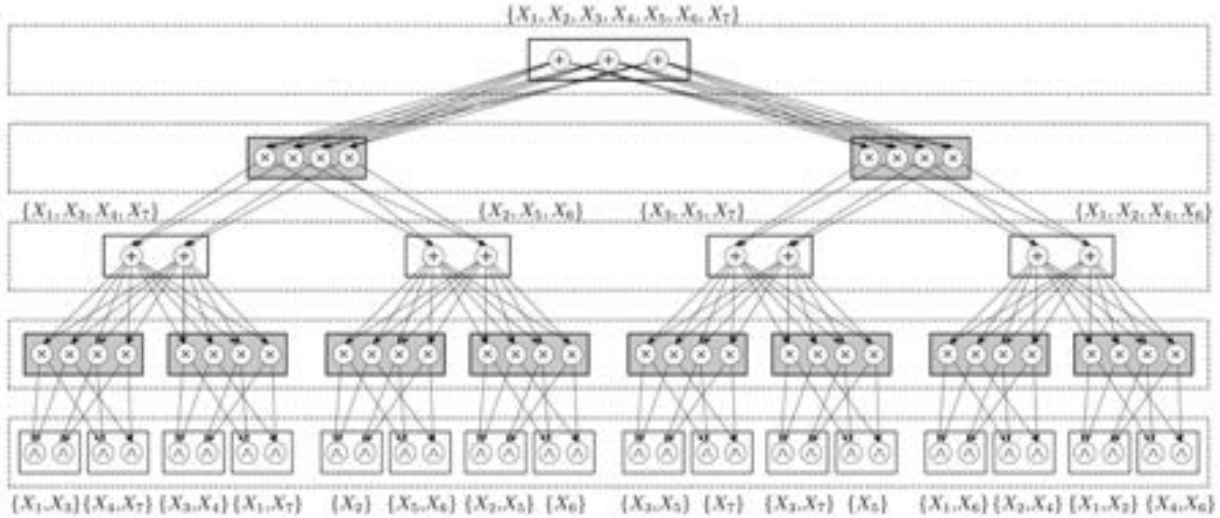
PERFORMANCE COMPARISON. BEST END-TO-END THROUGHPUTS (T), EXCLUDING THE CYCLE COUNTER MEASUREMENTS, ARE [illegible]TED BOLD.

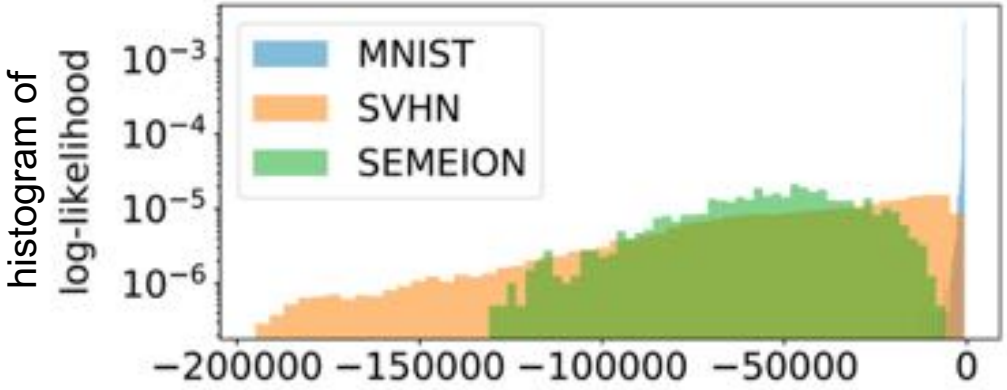
Dataset	Rows	CPU (µs)	T-CPU (rows/µs)	CPUF (µs)	T-CPUF (rows/µs)	GPU (µs)	T-GPU (rows/µs)	FPGA Cycle Counter	FPGAC (µs)	T-FPGAC (rows/µs)	FPG[illegible] (µs)	[illegible]FPGA (rows/µs)
Accidents	17009	2798.27	[illegible]	[illegible]	7.87	63090.94	0.27	17249	[illegible]	[illegible]	696.00	**24.44**
Audio	20000	4271.78	[illegible]	[illegible]	5.4[illegible]	[illegible]	[illegible]	20317	1[illegible]	[illegible]	761.00	**26.28**
Netflix	20000	4892.22	[illegible]	[illegible]	4.8[illegible]	[illegible]	[illegible]	20322	1[illegible]	[illegible]	654.00	**30.58**
MSNBC200	388434	15476.05	[illegible]	[illegible]	30.5[illegible]	[illegible]	[illegible]	388900	19[illegible]	[illegible]	008.00	**77.56**
MSNBC300	388434	10060.78	[illegible]	[illegible]	41.2[illegible]	[illegible]	[illegible]	388810	19[illegible]	[illegible]	933.00	**78.74**
NLTCS	21574	791.80	[illegible]	[illegible]	31.3[illegible]	[illegible]	[illegible]	21904	1[illegible]	[illegible]	566.00	**38.12**
Plants	23215	3621.71	6.41	3521.04	6.59	67004.41	0.35	23592	117.96	196.80	778.00	**29.84**
NIPS5	10000	25.11	**398.31**	26.37	379.23	8210.32	1.22	10236	51.18	195.39	337.30	[illegible]
NIPS10	10000	83.60	**119.61**	84.39	118.49	11550.82	0.87	10279	51.40	194.57	464.30	21.54
NIPS20	10000	191.30	52.27	182.73	**54.72**	18689.04	0.54	10285	51.43	194.46	543.60	18.40
NIPS30	10000	387.61	25.80	349.84	**28.58**	25355.93	0.39	10308	51.80	193.06	592.30	16.88
NIPS40	10000	551.64	18.13	471.26	**21.22**	30820.49	0.32	10306	51.53	194.06	632.20	15.82
NIPS50	10000	812.44	12.31	792.13	12.62	36355.60	0.28	10559	52.80	189.41	720.60	**13.88**
NIPS60	10000	1046.38	9.56	662.53	**15.09**	40778.36	0.25	12271	61.36	162.99	799.20	12.51
NIPS70	10000	1148.17	8.71	1134.80	8.81	46759.26	0.21	14022	70.11	142.63	858.60	**11.65**
NIPS80	10000	1556.99	6.42	1277.81	7.83	63217.99	0.16	14275	78.51	127.37	961.80	**10.40**

And also learning is conceptually easy

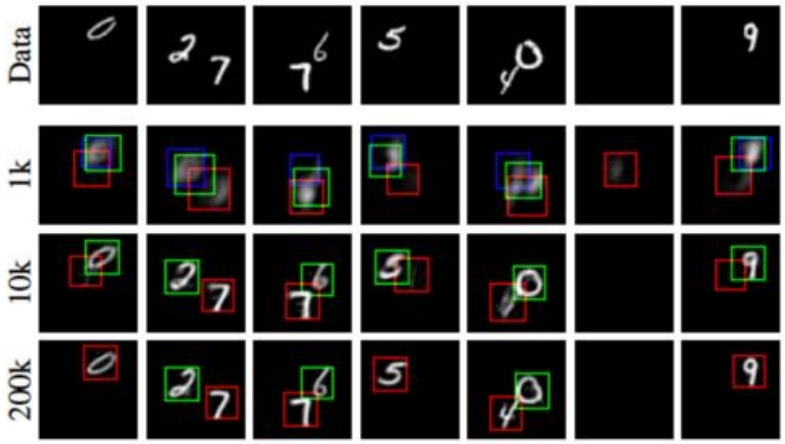
[Peharz, Vergari, Molina, Stelzner, Trapp, Kersting, Ghahramani UDL@UAI 2018]

Random sum-product networks



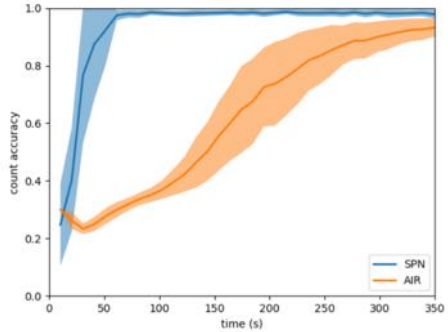


And also learning is conceptually easy

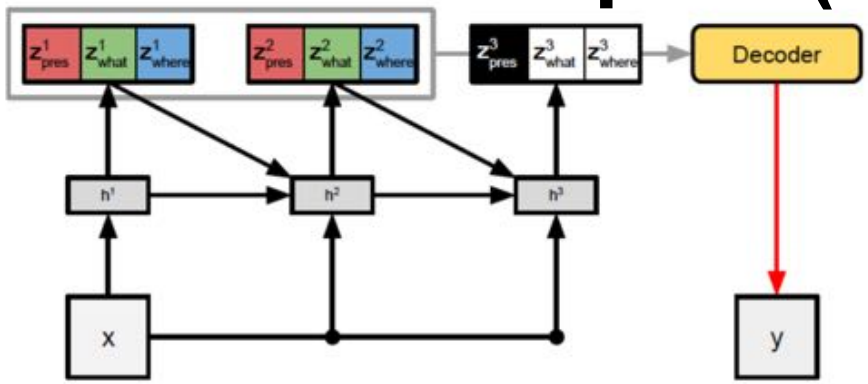


Explicit AIR

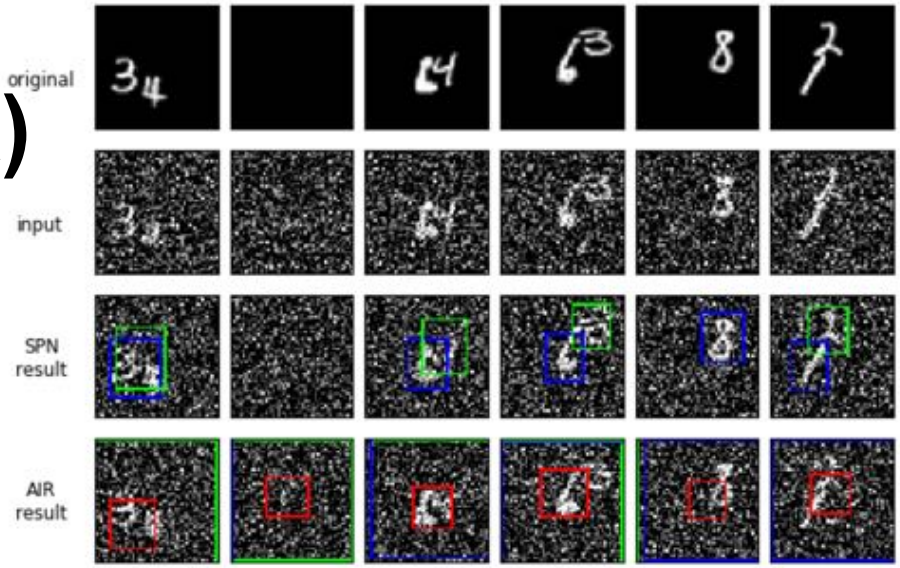
[Stelzner, Peharz, Kersting 2018]



Attend Infer Repeat (AIR)



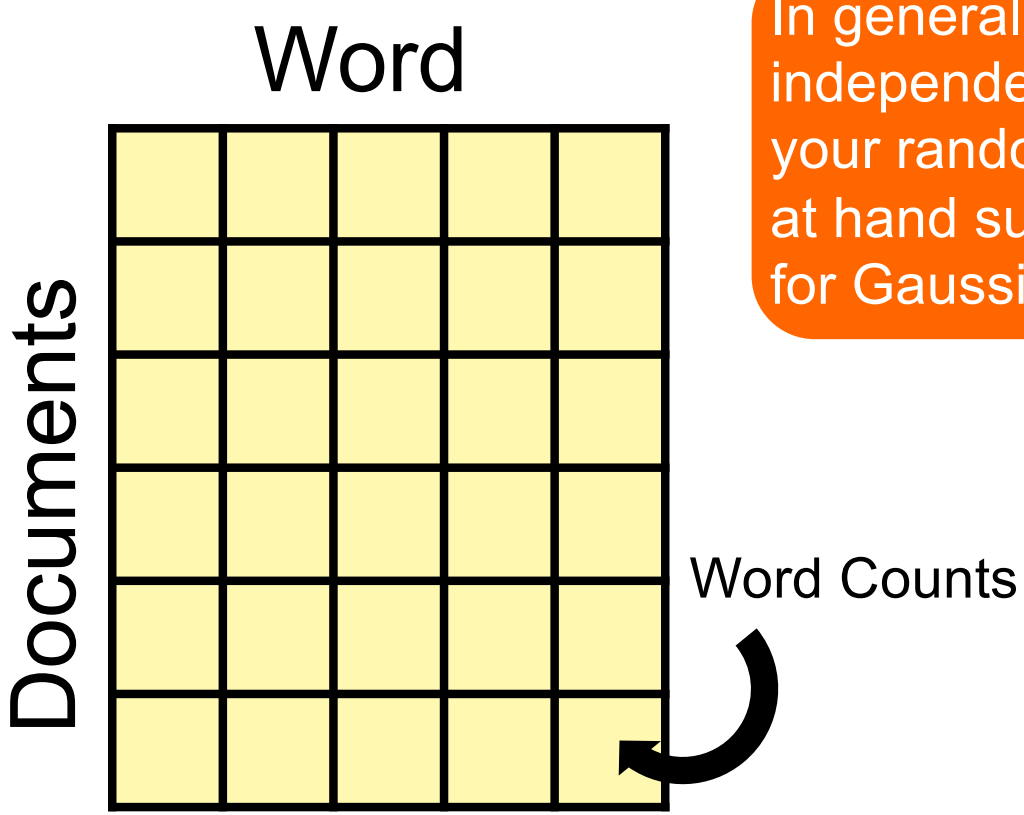
[Eslami et al. NIPS 2016]



Or you do greedy learning

TECHNISCHE UNIVERSITÄT DARMSTADT

Testing independence of random variables using e.g. nonparametric tests



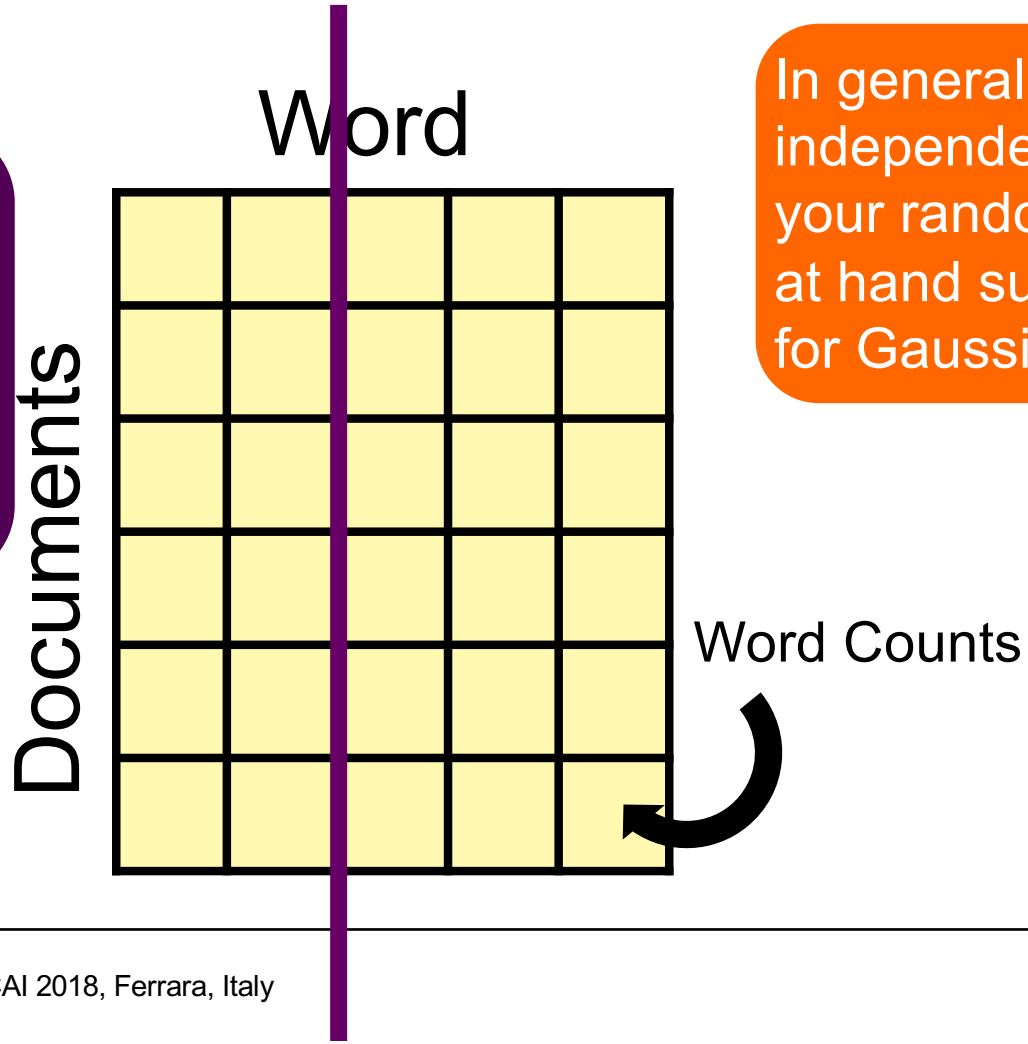
In general use the independency test for your random variables at hand such as g-test for Gaussians

Or you do greedy learning

TECHNISCHE UNIVERSITÄT DARMSTADT

Testing independence of random variables using e.g. nonparametric tests

Learn GLM model trees for P(x|V-x) and P(y|V-y). Check whether X resp. Y is significant in P(y|V-x) resp. P(x|V-y)

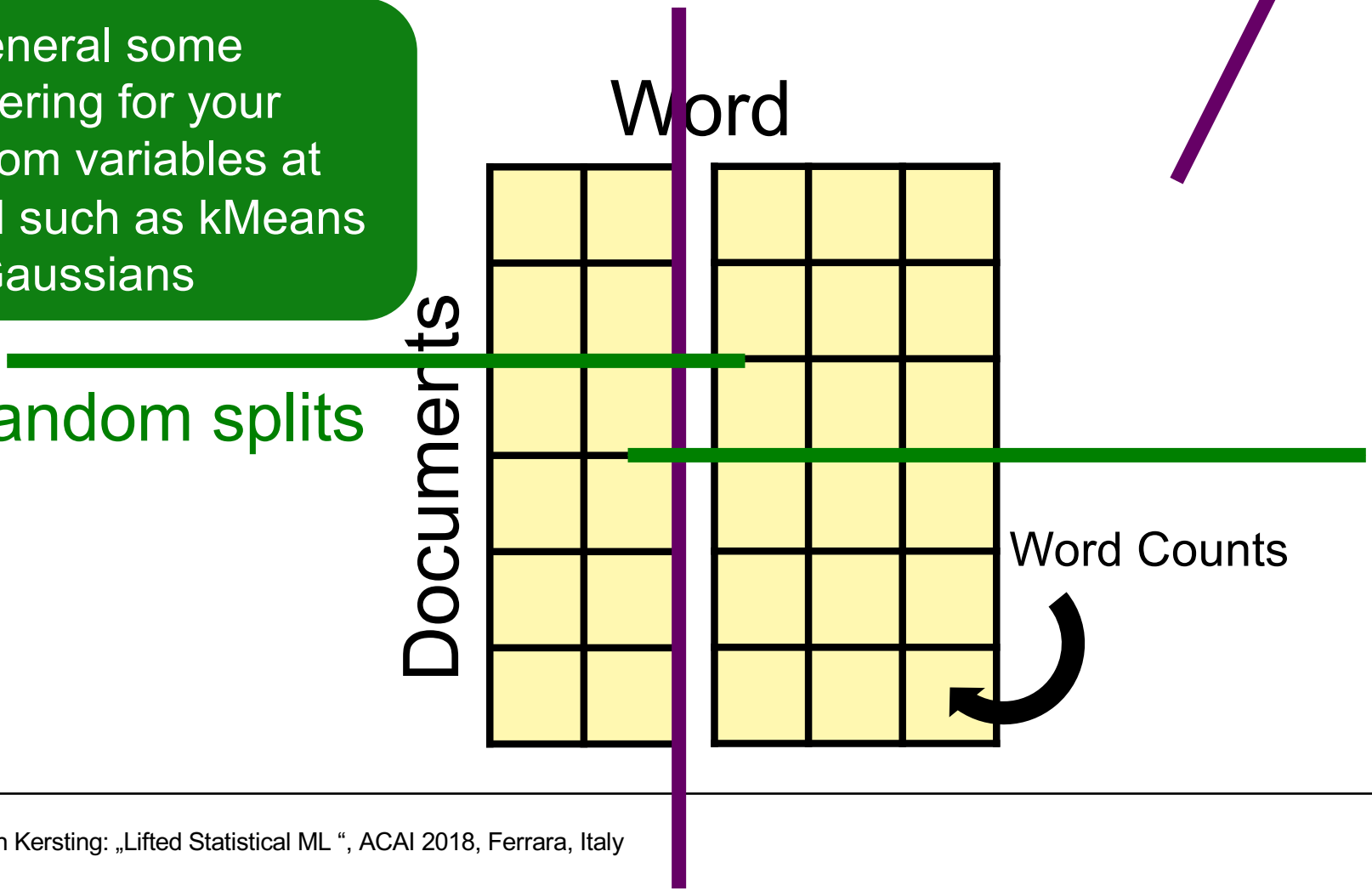


In general use the independency test for your random variables at hand such as g-test for Gaussians

Or you do greedy learning

Testing independence of random variables using e.g. nonparametric tests

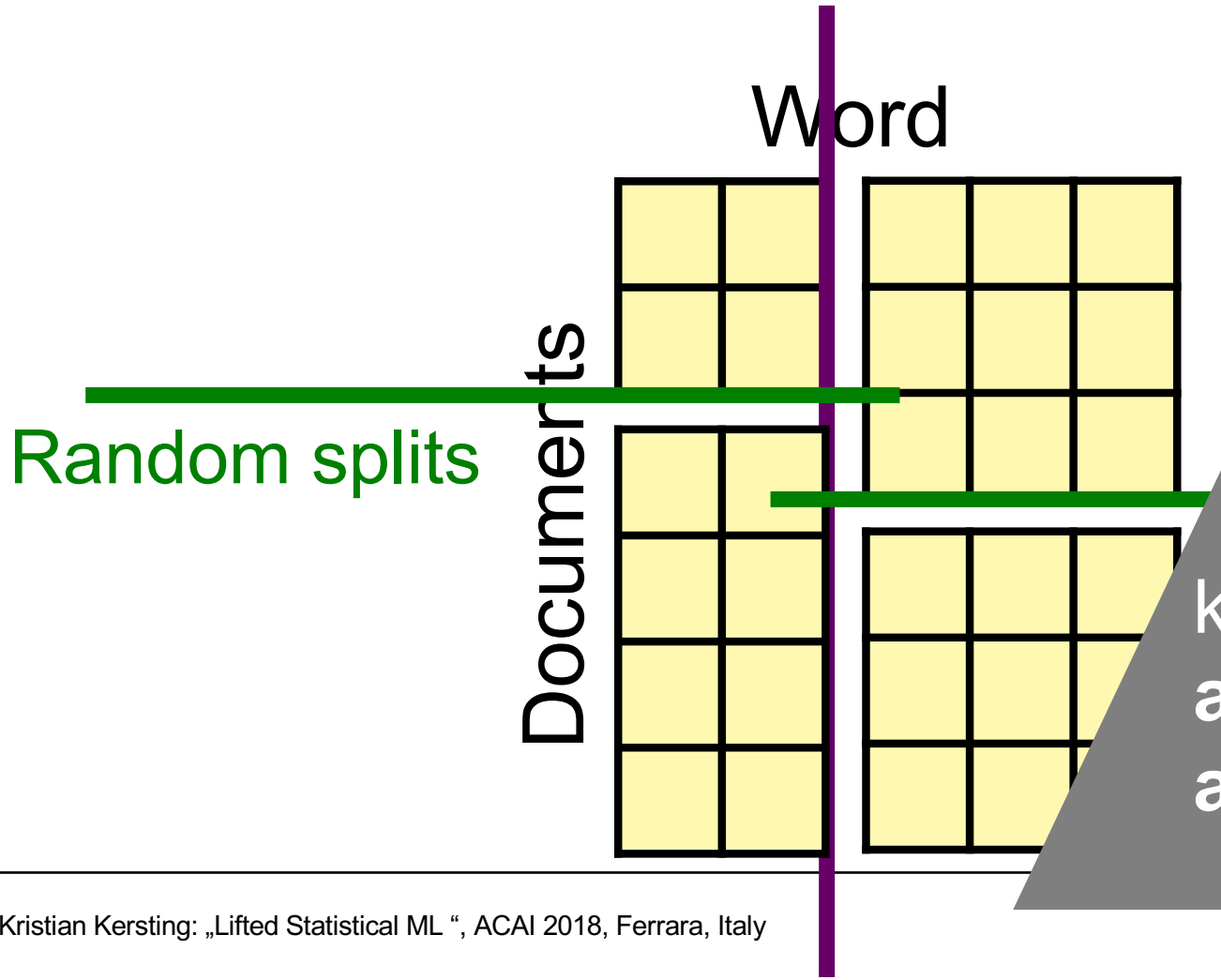
In general some clustering for your random variables at hand such as kMeans for Gaussians

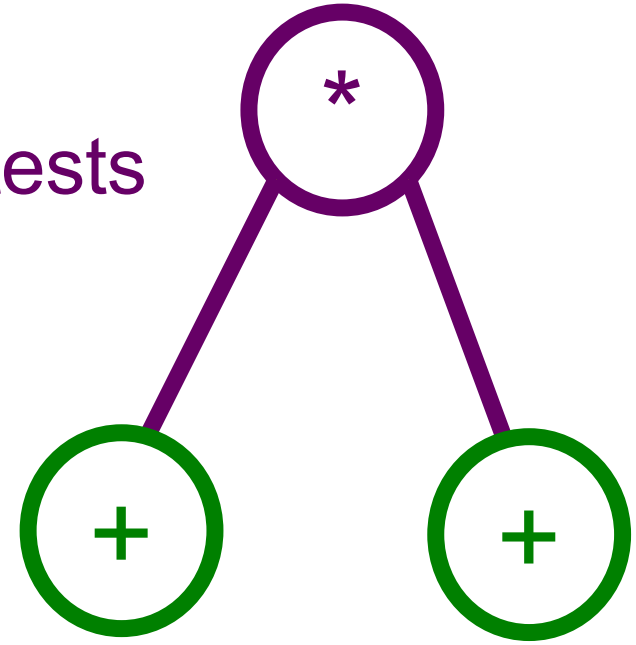


[Poon, Domingos UAI'11; Molina, Natarajan, Kersting AAAI`17, Molina et al AAAI ,18]

Or you do greedy learning

Testing independence of random variables using e.g. nonparametric tests

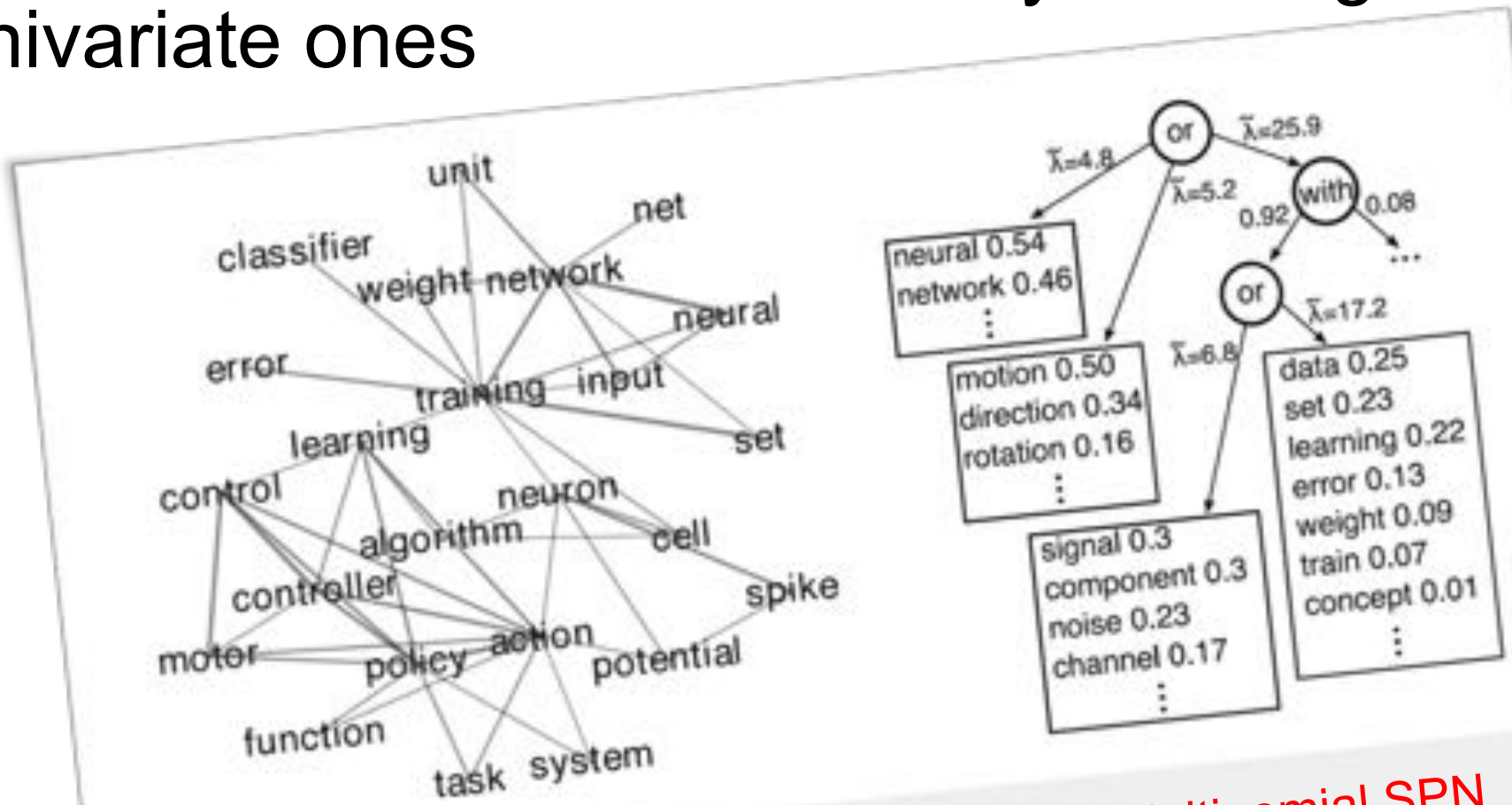




keep growing alternatingly and + layers *

[Molina, Natarajan, Kersting AAAI 2017]

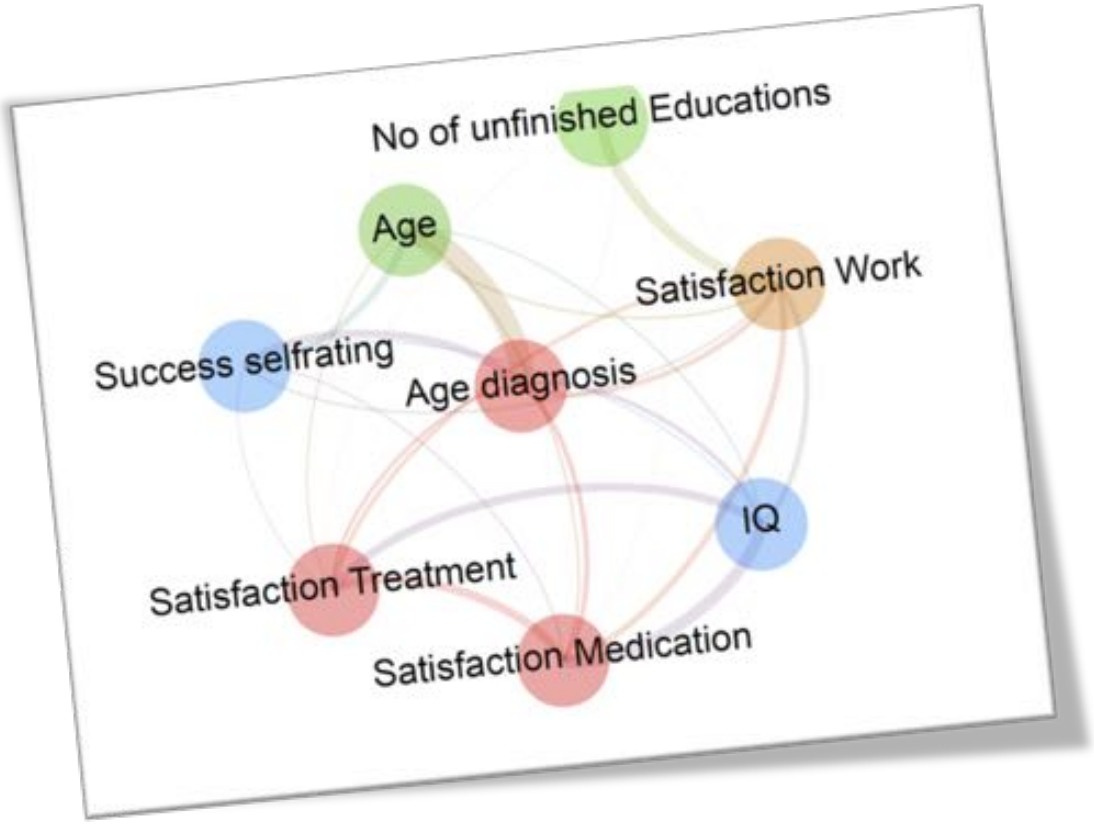
Probabilistic modelling made easy: Build multivariate distribution by stacking univariate ones



Poisson Mutual Information (NIPS corpus)

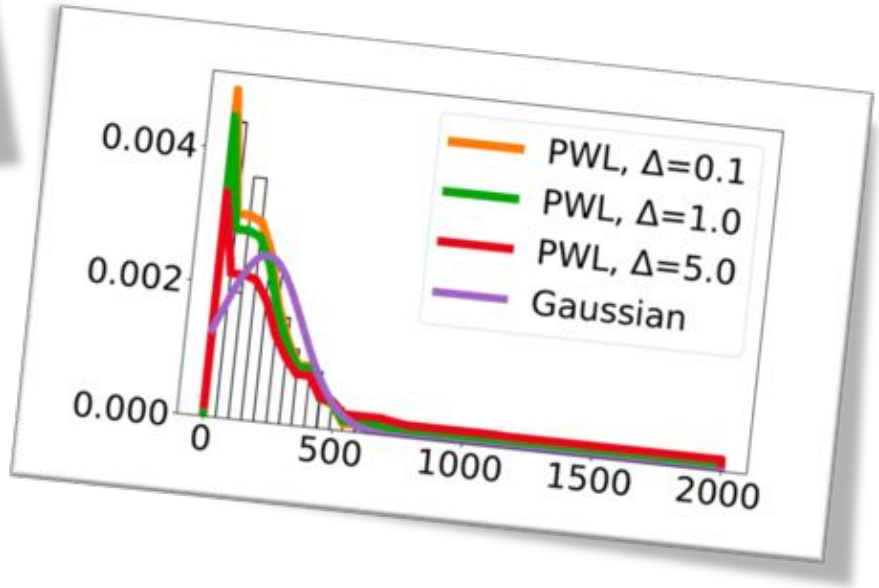
Poisson Multinomial SPN = hierachical topic model

… even in a distribution-agnostic way



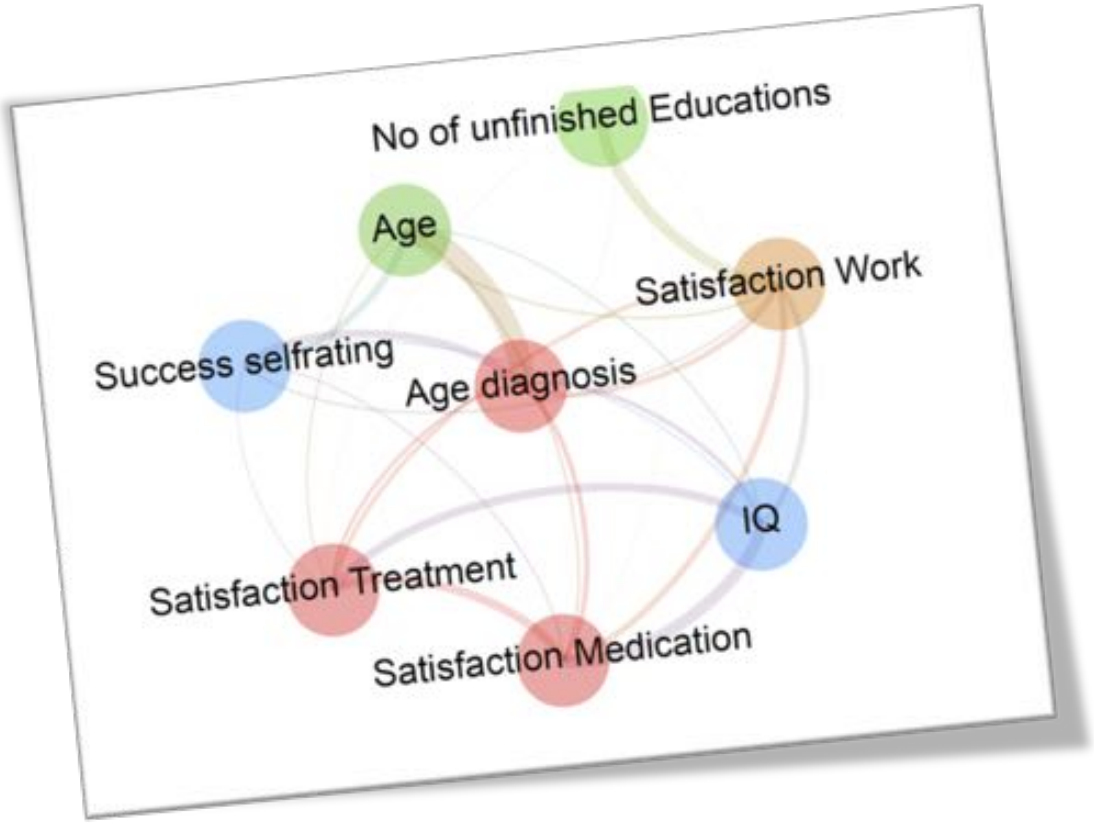
Visualization the gist of a Mixed SPN (learned on the hybrid Autism dataset) using normalized mutual info (the thicker, the higher). **The machine does not know anything about the data types.**

Use nonparametric independency tests and piece-wise linear approximations



[Molina, Natarajan, Vergari, Di Mauro, Esposito, Kersting AAAI 2018]

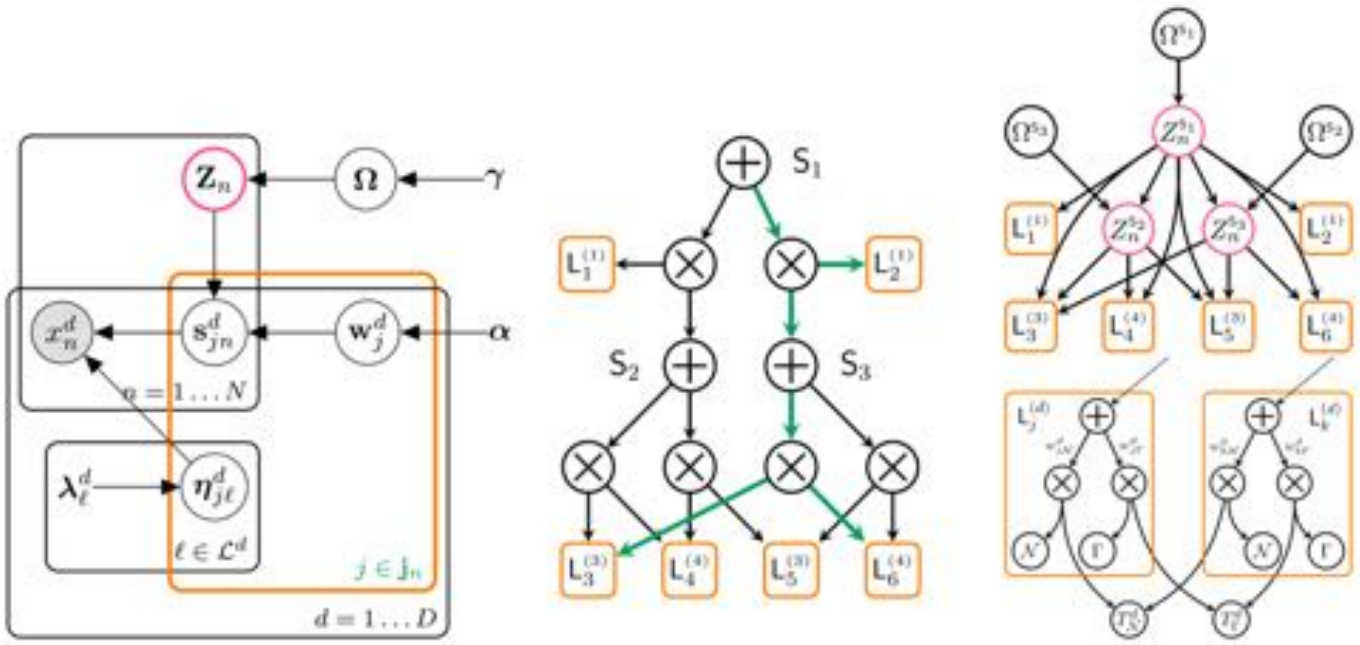
… even in a distribution-agnostic way



Visualization the gist of a Mixed SPN (learned on the hybrid Autism dataset) using normalized mutual info (the thicker, the higher). **The machine does not know anything about the data types.**

However, we have to provide the statistical types and do not gain insights into the parametric forms of the variables. **Are they Gaussians? Gammas? …**

Automatic Bayesian Density Analysis



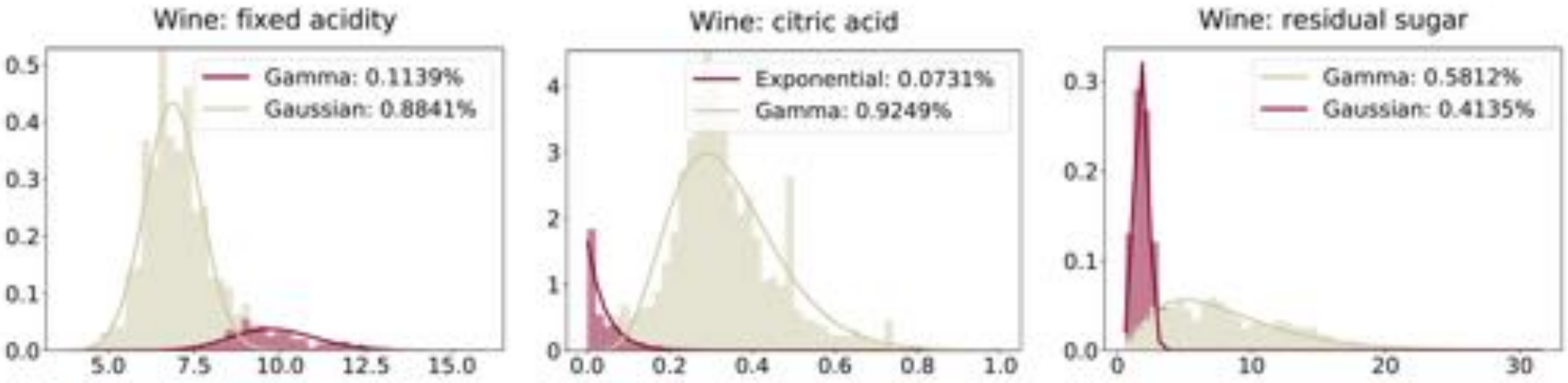
(a) Graphical model (b) SPN (c) Type-augmented SPN

Bayesian discovery of statistical types and parametric forms of variables

+

Type-agnostic deep probabilistic learning

Automatic Bayesian Density Analysis



… can automatically discovers the statistical types and parametric forms of the variables

Automatic Bayesian Density Analysis

	transductive setting						*inductive setting*	
	10%			50%			70%-10%-20%	
	ISLV	**ABDA**	**MSPN**	**ISLV**	**ABDA**	**MSPN**	**ABDA**	**MSPN**
Abalone	-1.15±0.12	-0.02±0.03	**0.20**	-0.89±0.36	-0.05±0.02	**0.14**	2.22±0.02	**9.73**
Adult	-	**-0.60**±0.02	-3.46	-	**-0.69**±0.01	-5.83	**-5.91**±0.01	-44.07
Australian	-7.92±0.96	**-1.74**±0.19	-3.85	-9.37±0.69	**-1.63**±0.04	-3.76	**-16.44**±0.04	-36.14
Autism	-2.22±0.06	**-1.23**±0.02	-1.54	-2.67±0.16	**-1.24**±0.01	-1.57	**-27.93**±0.02	-39.20
Breast	-3.84±0.05	-2.78±0.07	**-2.69**	-4.29±0.17	**-2.85**±0.01	-3.06	**-25.48**±0.05	-28.01
Chess	-2.49±0.04	**-1.87**±0.01	-3.94	-2.58±0.04	**-1.87**±0.01	-3.92	**-12.30**±0.00	-13.01
Crx	-12.17±1.41	**-1.19**±0.12	-3.28	-11.96±1.01	**-1.20**±0.04	-3.51	**-12.82**±0.07	-36.26
Dermatology	-2.44±0.23	**-0.96**±0.02	-1.00	-3.57±0.32	**-0.99**±0.01	-1.01	**-24.98**±0.19	-27.71
Diabetes	-10.53±1.51	**-2.21**±0.09	-3.88	-12.52±0.52	**-2.37**±0.09	-4.01	**-17.48**±0.05	-31.22
German	-3.49±0.21	**-1.54**±0.01	-1.58	-4.06±0.28	**-1.55**±0.01	-1.60	**-25.83**±0.05	-26.05
Student	-2.83±0.27	**-1.56**±0.03	-1.57	-3.80±0.29	**-1.57**±0.01	-1.58	**-28.73**±0.10	-30.18
Wine	-1.19±0.02	-0.90±0.02	**-0.13**	-1.34±0.01	-0.92±0.01	**-0.41**	-10.12±0.01	**-0.13**
wins	0	9	3	0	10	2	10	2

… but also models its uncertainty about the statistical types and parametric forms, which can lead to better models

The machine understands the data with few expert input …

Toggle Introduction

Toggle explanations

Toggle Code

Völker: “DeepNotebooks – Interactive data analysis using Sum-Product Networks.“ MSc Thesis, TU Darmstadt, 2018

Exploring the Titanic dataset

This report describes the dataset Titanic and contains general statistical information and an analysis on the influence different features and subgroups of the data have on each other. The first part of the report contains general statistical information about the dataset and an analysis of the variables and probability distributions. The second part focusses on a subgroup analysis of the data. Different clusters identified by the network are analyzed and compared to give an insight into the structure of the data. Finally the influence different variables have on the predictive capabilities of the model are analyzes.
The whole report is generated by fitting a sum product network to the data and extracting all information from this model.

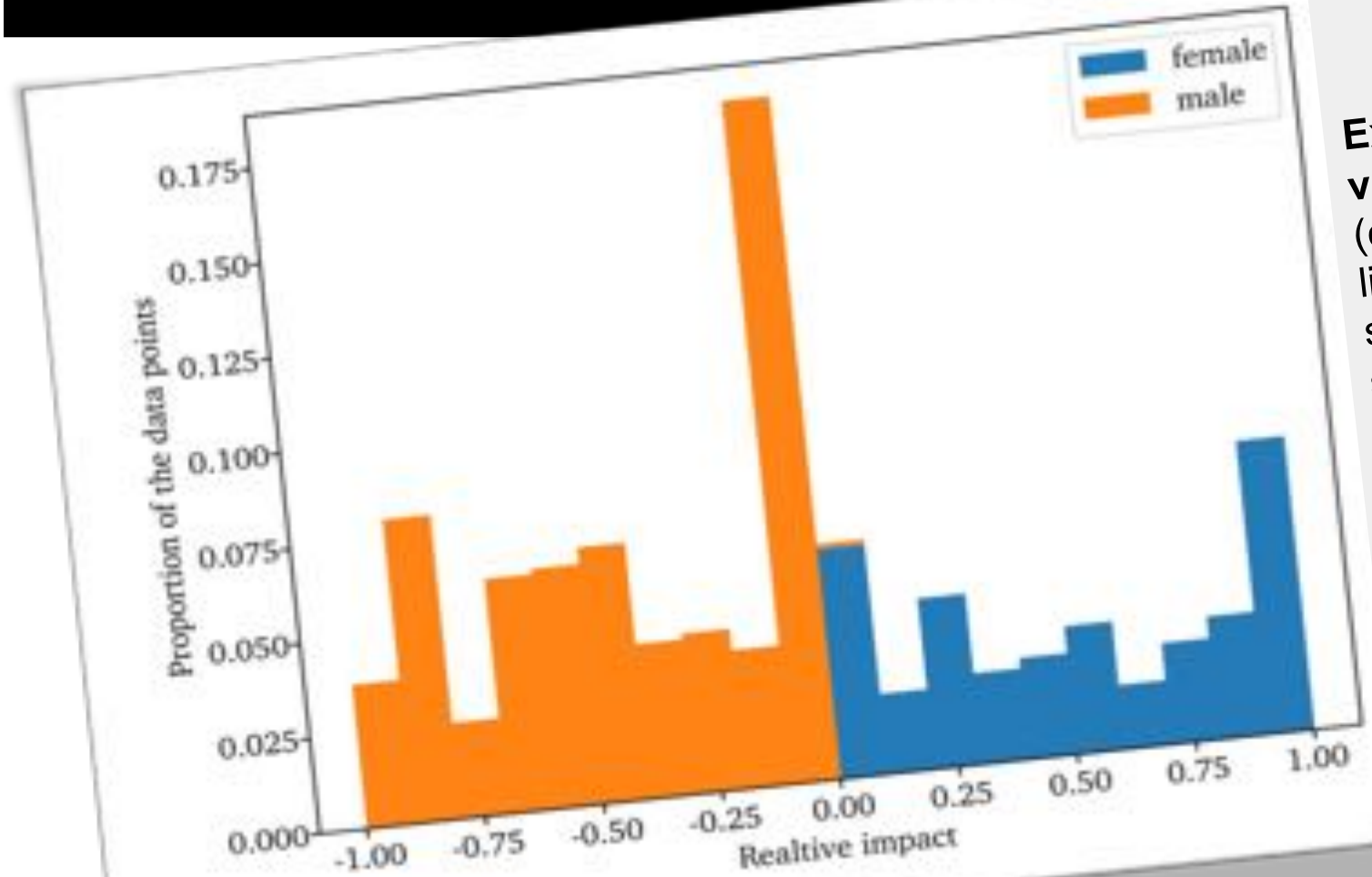
TECHNISCHE UNIVERSITAT DARMSTADT

Report framework created @ TU Darmstadt

…and can compile data reports automatically

*[Baehrens, Schroeter, Harmeling, Kawanabe, Hansen, Müller JMLR 11:1803-1831, 2010]

The machine understands the data with no expert input …



Explanation vector* (computable in linear time in the sizre of the SPN) **showing the impact of "gender" on the chances of survival for the Titanic dataset**

…and can compile data reports automatically

What have we learnt about SPNs?

TECHNISCHE UNIVERSITÄT DARMSTADT

Sum-product networks (SPNs)

- DAG of sums and products
- They are instances of Arithmetic Circuits (ACs)
- Compactly represent partition function
- Learn many layers of hidden variables

Efficient marginal inference

Easy learning

Can outperform well-known alternatives

Take-away messges of Part I

- Graphical models are great to deal with probability distributions
- To make them „tractable“ we can employ symmetries
- Or we build directly „tractable“ graphical models, i.e., computation graphs as in TensorFlow but with probabilistic semantics
- This can lead to inference on the device and can free the user from making assumptions on the statistical form of the data

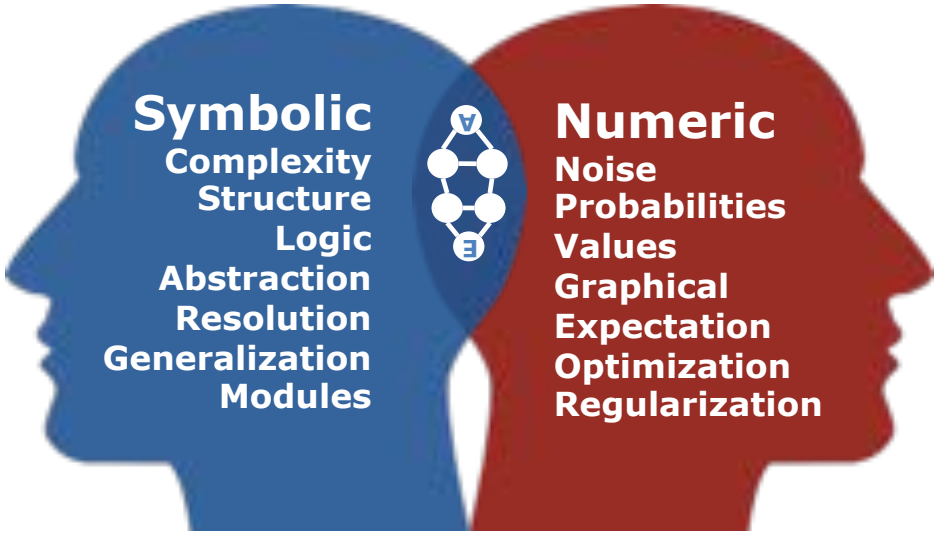
Lifted Statistical Machine Learning

Computational modeling of complex AI systems that learn and think

Part II: Statistical Relational AI

Kristian Kersting

Thanks to Babak Ahmadi, Vincent Conitzer, Rina Dechter, Luc De Raedt, Pedro Domingos, Peter Flach, Dieter Fensel, Florian Ficher, Vibhav Gogate, Carlos Guestrin, Daphen Koller, Nir Friedman, Martin Mladenov, Ray Mooney, Sriraam Natarajan, David Poole, Fabrizio Riguzzi, Dan Suciu, Guy van den Broeck, and many others for making their slides publically available



Goals of Part II

Get in touch with

1. Statistical Relational Learning and Probabilistic Programming, and
2. understand that this covers the whole AI spectrum, leading to Systems AI

Caution! Necessarily incomplete!

Let's consider a simple relational domain: Reviewing Papers

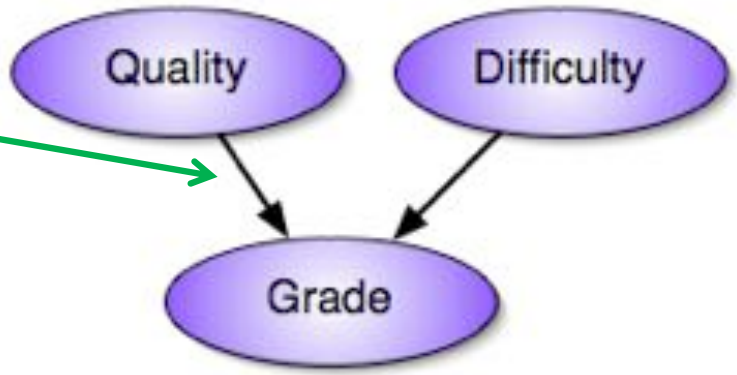
The grade of a paper at a conference depends on the paper's quality and the difficulty of the conference.

- **Good papers may get A's at easy conferences**
- **Good papers may get D's at top conference**
- **Weak papers may get B's at good conferences**
- **…**

(Reviewing) Bayesian Network

Random Variables

Direct Influence



P(Qual)		
low	middle	high
0.3	0.5	0.2

P(Diff)		
low	middle	high
0.2	0.3	0.5

$$P(X_1,\ldots,X_n)=\prod_{i=1}^{n} P(X_i \mid X_{i-1},\ldots,X_1)$$

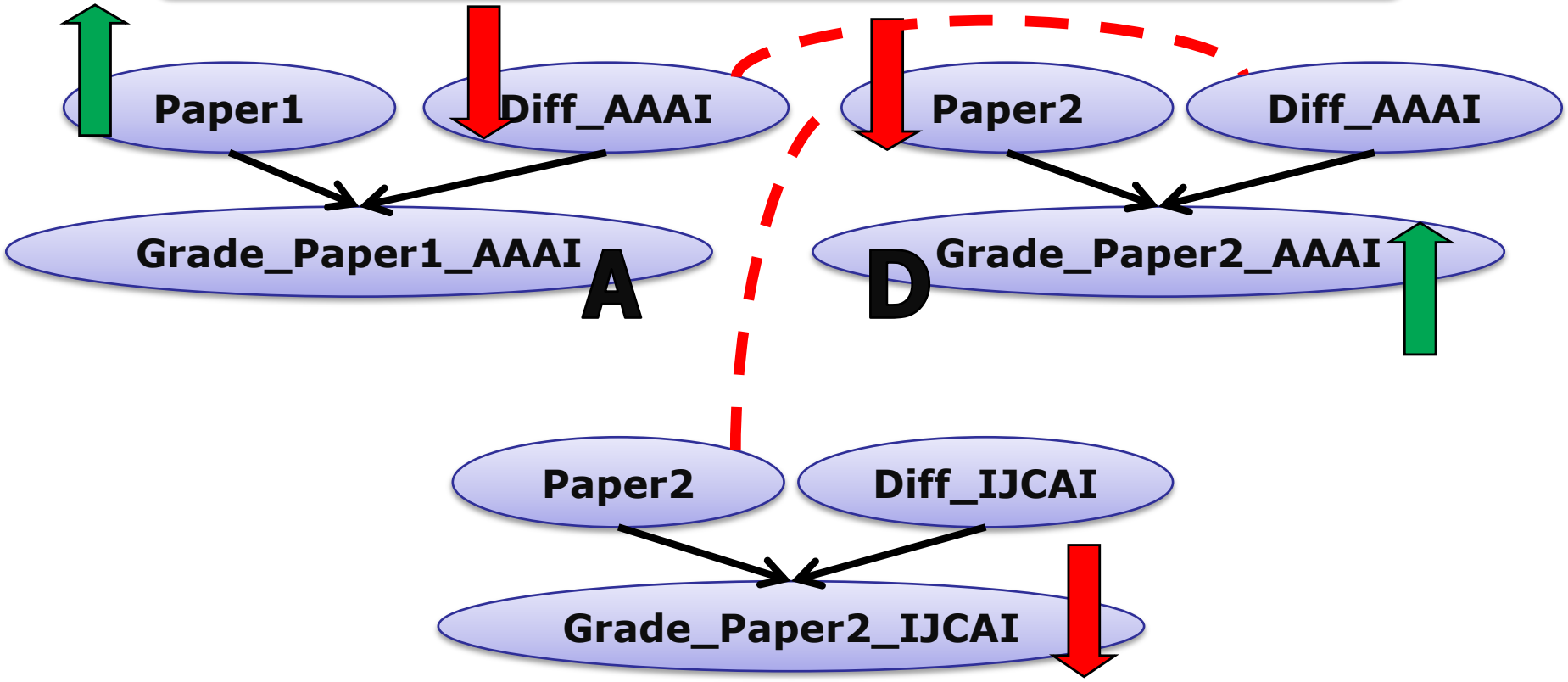
Qual	Diff	P(Grade)		
		c	b	a
low	low	0.2	0.5	0.3
low	middle	0.1	0.7	0.2
...				

The real world, however, …

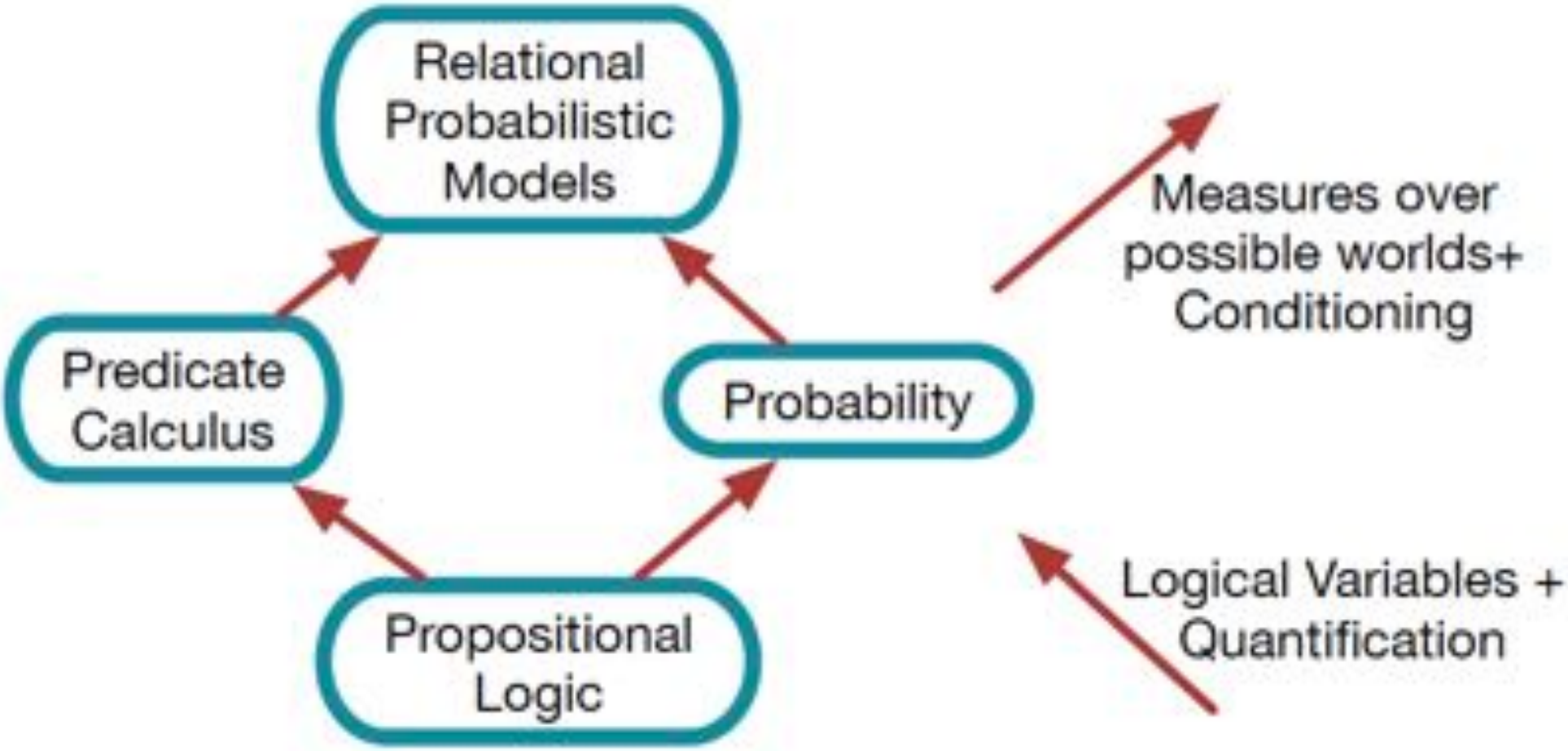
[inspired by Friedman and Koller]

… has inter-related objects

These 'instance' are not independent !



Therefore we want to combine probabilities and logic

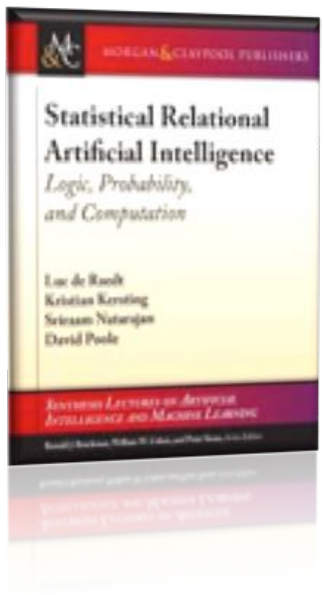


Systems AI: the computational and mathematical modeling of complex AI systems.

Eric Schmidt, Executive Chairman, Alphabet Inc.: Just Say "Yes", Stanford Graduate School of Business, May 2, 2017.https://www.youtube.com/watch?v=vbb-AjiXyh0. But also see e.g. Kordjamshidi, Roth, Kersting: “Systems AI: A Declarative Learning Based Programming Perspective.“ IJCAI-ECAI 2018.

For Systems AI we have to deeply understand data, knowledge and reasoning in a large number of forms

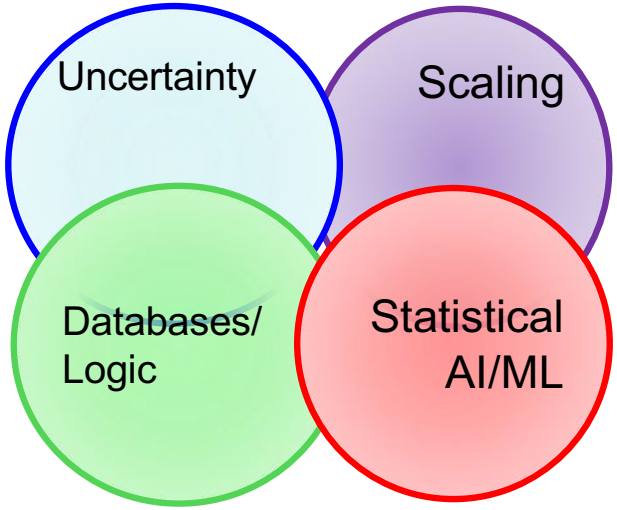
Crossover of Statistical AI/ML with data & programming abstractions



building general-purpose thinking and learning machines

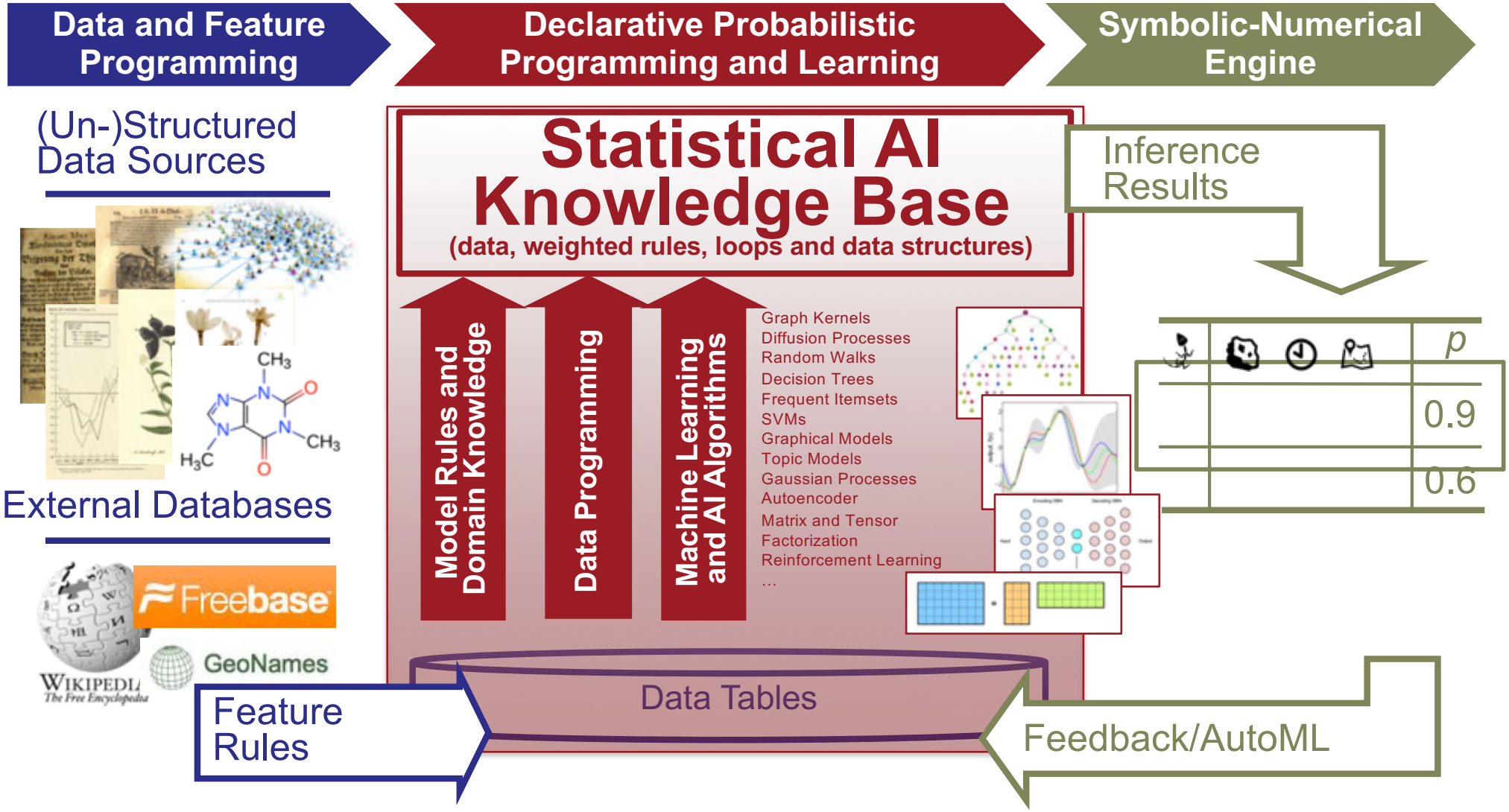
make the AI/ML expert more effective

increases the number of people who can successfully build AI/ML applications



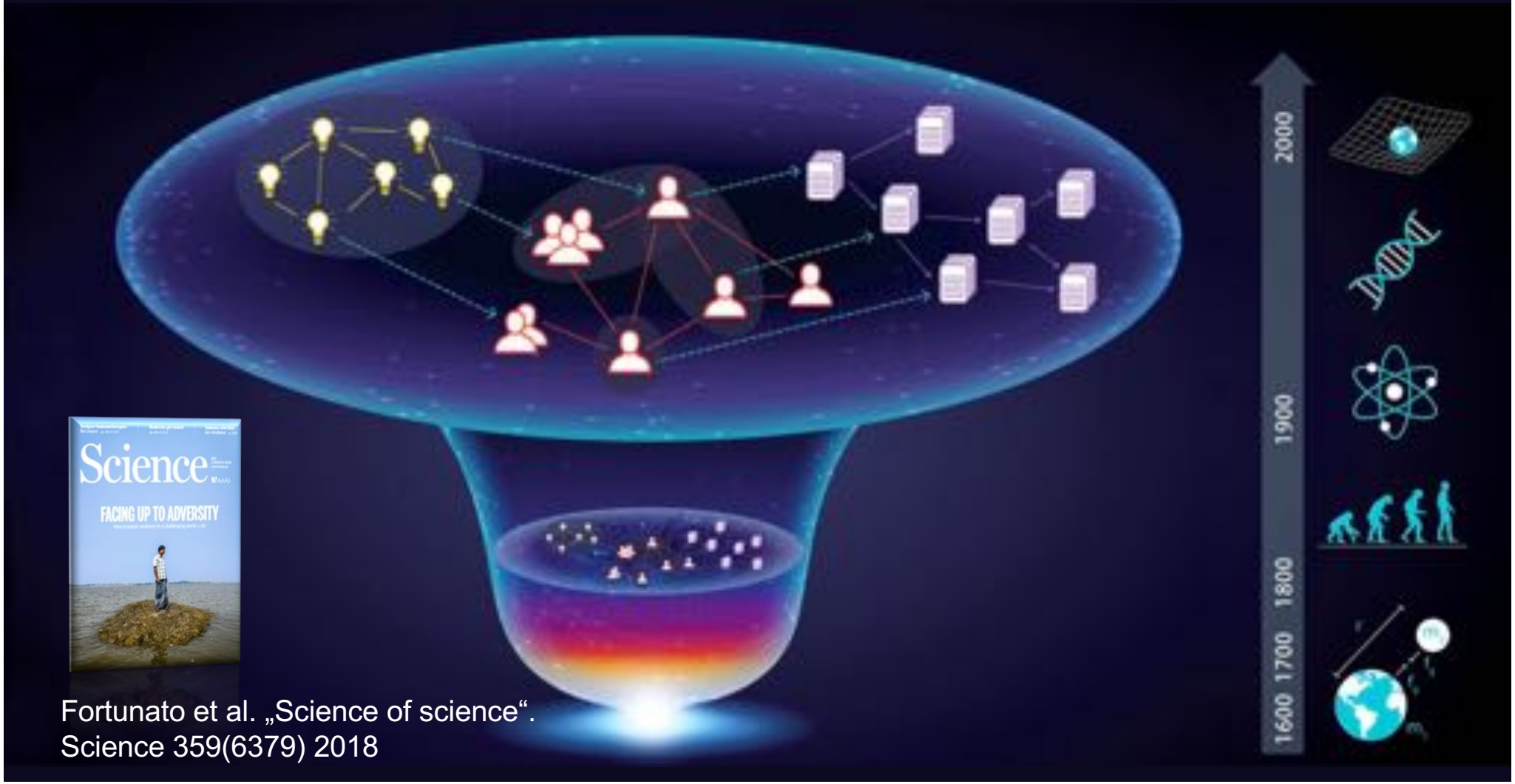
De Raedt, Kersting, Natarajan, Poole: Statistical Relational Artificial Intelligence: Logic, Probability, and Computation. Morgan and Claypool Publishers, ISBN: 9781627058414, 2016.

This establishes a novel "Deep AI"



[Ré et al. IEEE Data Eng. Bull.'14; Natarajan, Picado, Khot, Kersting, Ré, Shavlik ILP'14; Natarajan, Soni, Wazalwar, Viswanathan, Kersting Solving Large Scale Learning Tasks'16, Mladenov, Heinrich, Kleinhans, Gonsior, Kersting DeLBP'16, …]

Machine Learning can be seen as an expanding and evolving network of ideas, scholars, and papers. Can machines read this data?



Fortunato et al. „Science of science“.
Science 359(6379) 2018

Benjamin generates the ML Genome

[Bratzadeh. MSc Thesis TU Dortmund 2016]

The Machine Learning (ML) Genome is a dataset, a knowledge base, an ongoing effort to learn and reason about ML concepts

Algorithms

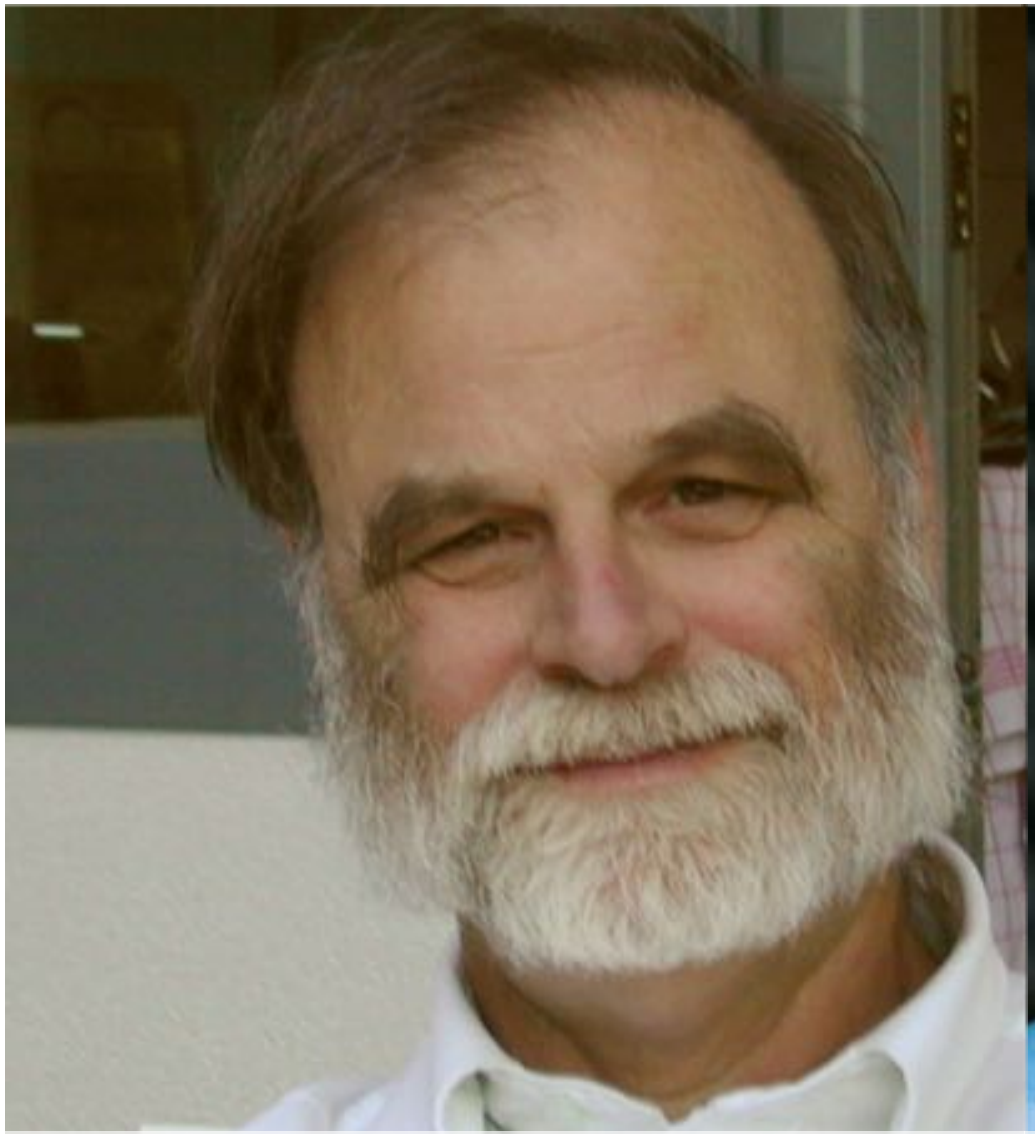
Compared to

Introduced algorithms are mentioned in the abstract

Algorithms mentioned in experiments have been compared to the introduced one

And connects well to database theory

TECHNISCHE UNIVERSITÄT DARMSTADT



Jim Gray Turing Award 1998
"Automated Programming"

Mike Stonebraker Turing Award 2014
"One size does not fit all"

… and cognitive science

"How do we humans get so much from so little?" and by that I mean how do we acquire our understanding of the world given what is clearly by today's engineering standards so little data, so little time, and so little energy.

Josh Tenenbaum
“Bayesian Program Learning”

Lake, Salakhutdinov, Tenenbaum, Science 350 (6266), 1332-1338, 2015
Tenenbaum, Kemp, Griffiths, Goodman, Science 331 (6022), 1279-1285, 2011

But first let us clarify how people view relations in this context

SO WHAT ARE RELATIONS?

What are Relations?

There are several types of relations and in turn there are several views on what (statitical) relational learning is

1. **Relations provide additional correlations/ regularization**
   - If two words occure frequently in the same context (page, paragraph, sentence, …) then there must be some semantic relation between them
2. **Often extensional (data) only, for one relation**
   - Covariance function, distance functions, kernel functions, graphs, tensors, random walks with restarts…

What are Relations?

TECHNISCHE UNIVERSITÄT DARMSTADT

3. **Relations are symmetries/redundancies in the model**
   - E.g. lifted inference based on bisimulation
4. **Hypergraph representations of data**
   - Multiple (extensional) relations
   - Random walks with restarts as similarity measure, produce path features, tensor-based embeddings
5. **Full-fledged relational (or logical) knoweldge as considered in this tutorial**
   - Multiple (often typed) relations
   - Intensional formulas (often Horn clauses) ancestor(X,Z) ^ parent(Z,Y) ⇒ ancestor(X,Y)

Over the years many SRL frameworks have been proposed:

THE SRL ALPHABET SOUP

Easy to miss the big picture but thankfully they all can be structured along some

KEY REPRESENTATION DIMENSIONS

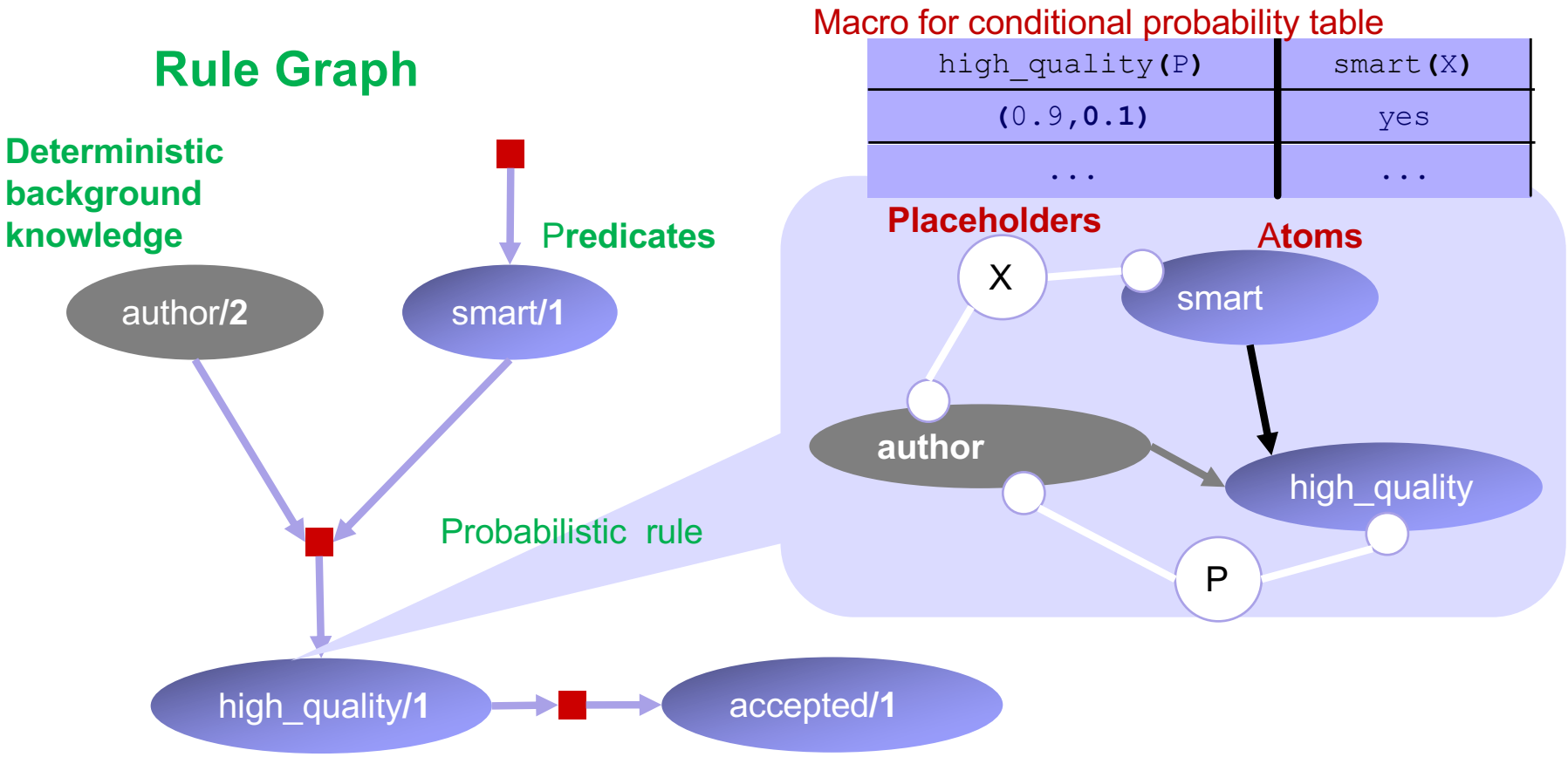
Key Dimensions with some prototypes

Directed: Probabilistic Relational Models (PRMs) Bayesian logic Programs (BLPs)

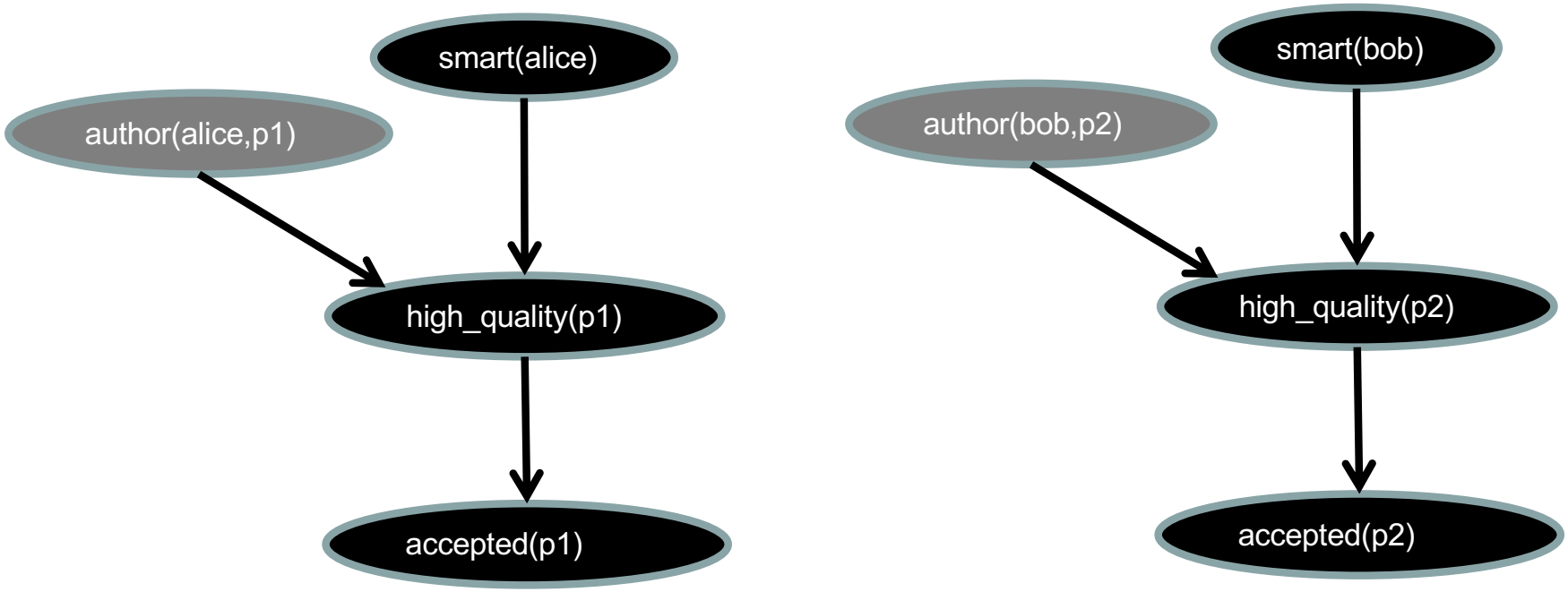
[Getoor et al. 2002; Kersting, De Raedt 2007]

$$\forall x\ author(x,p) \wedge smart(x) \Rightarrow high\_quality(p)$$

$$\forall x\ high\_quality(p) \Rightarrow accepted(p)$$

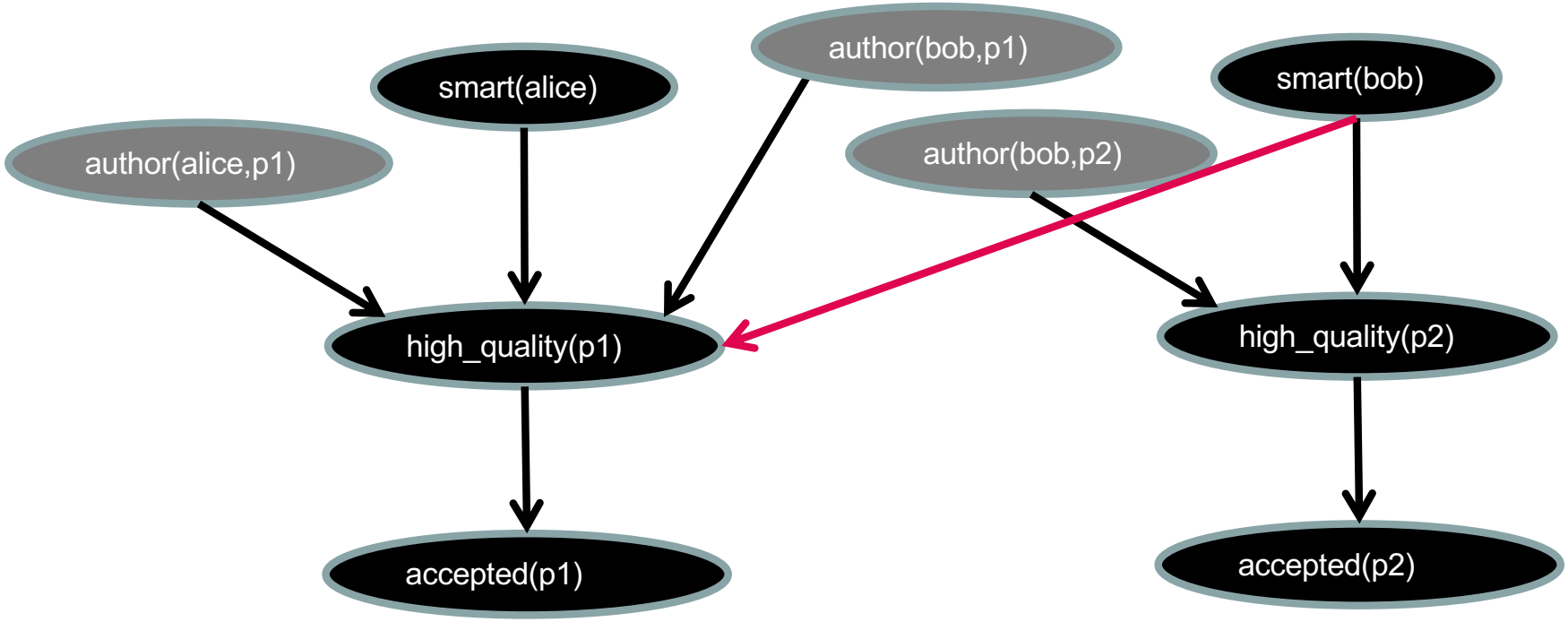


Inference on BN constructed by instantiating the rules/ macros using back- or forward chaining



So, we can deal with a variable number of objects. The induced BN depends on the domain elements and the background knowledge we have.

Inference on BN constructed by instantiating the rules/ macros using back- or forward chaining

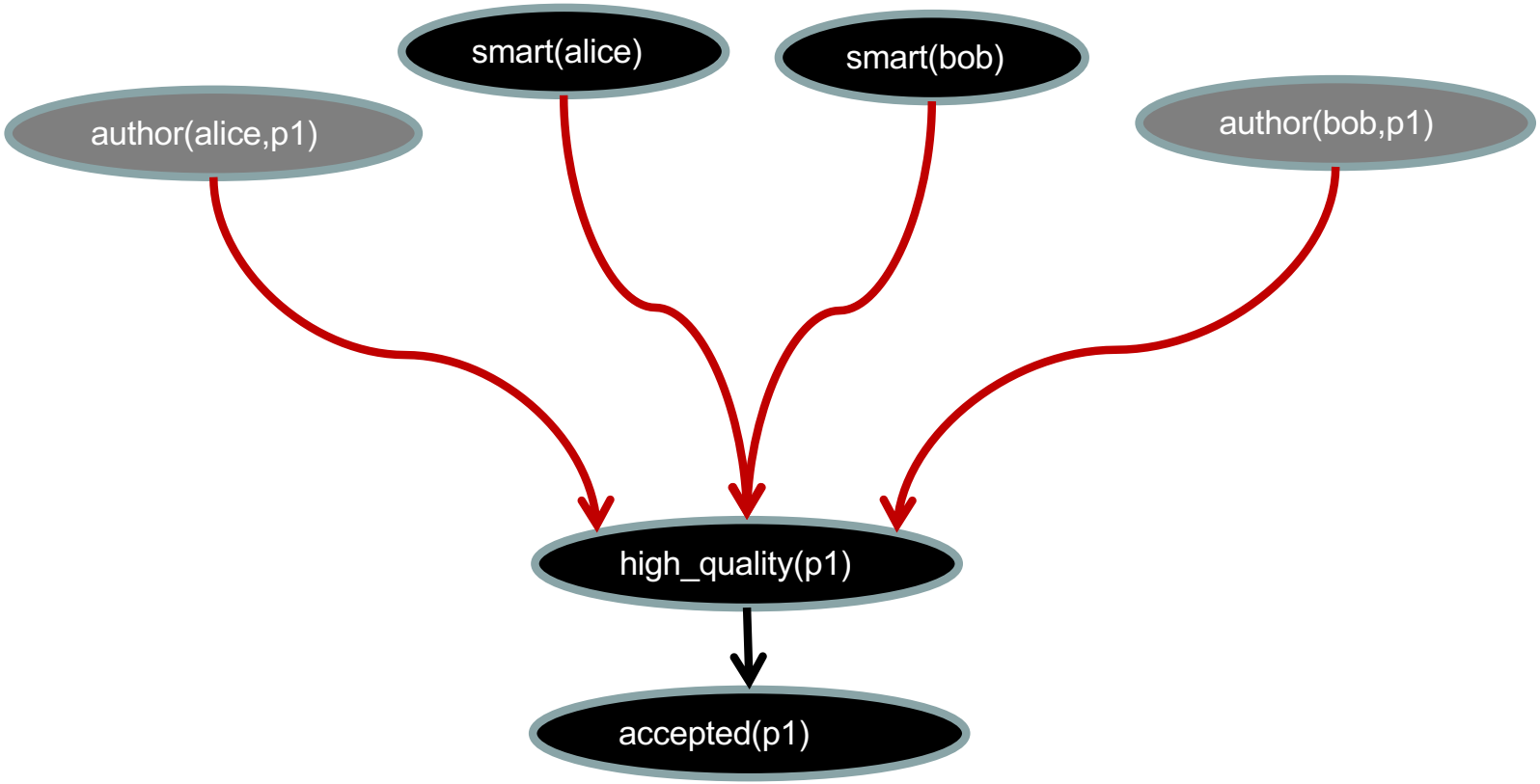


But what happens if we also have author(bob,p1)?

So, we can deal with a variable number of objects. The induced BN depends on the domain elements and the background knowledge we have.

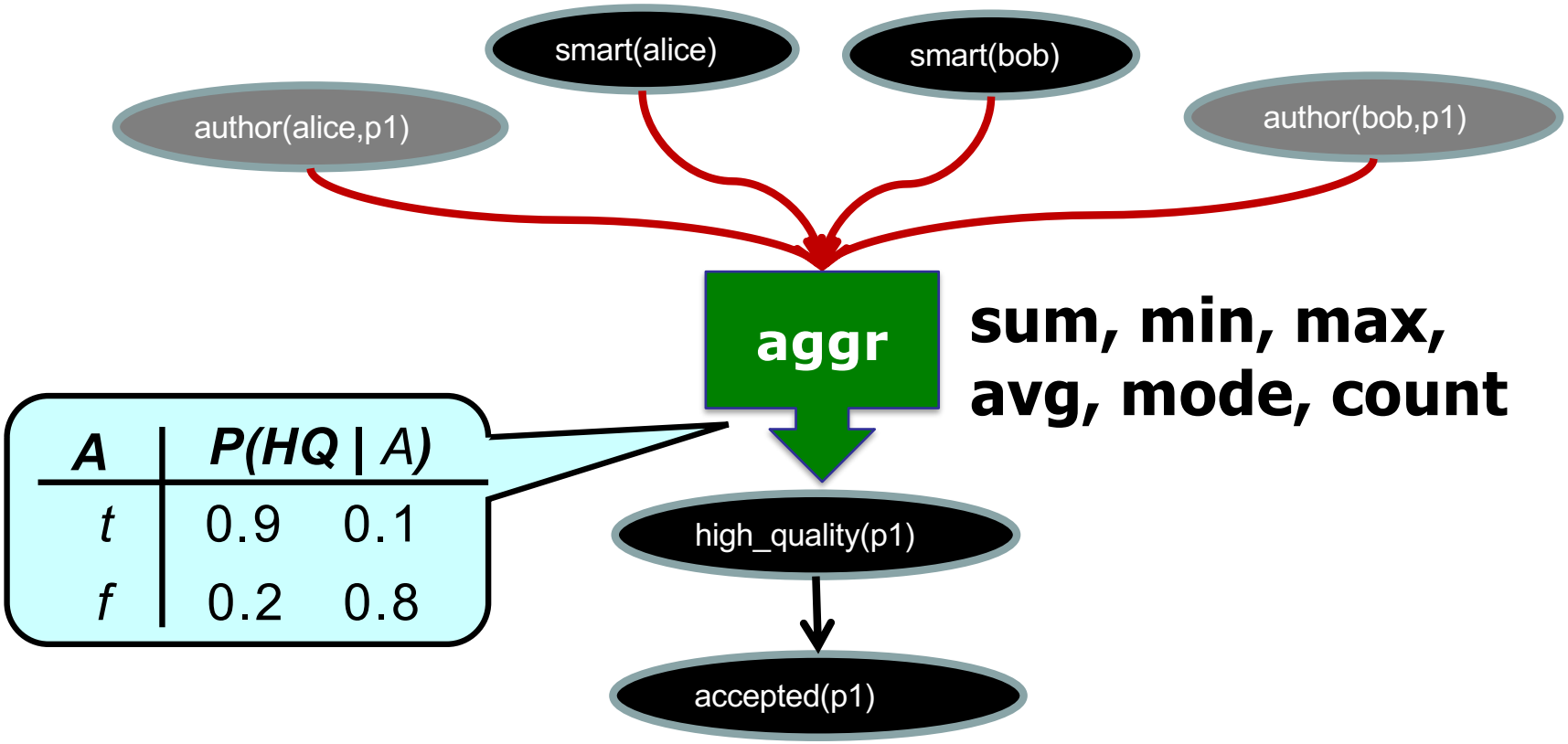
Directed: Aggregate Dependencies

We have several conditional probabilities instantiated from the same clause, which we have to aggregate



Directed: Aggregate Dependencies

We have several conditional probabilities instantiated from the same clause, which we have to aggregate



A	*P(HQ | A)*	
t	0.9	0.1
f	0.2	0.8

Still, the induced model is assumed to be acyclic

[Neville, Jensen 2007]

Option 1 : Relational Dependency Networks (RDNs)

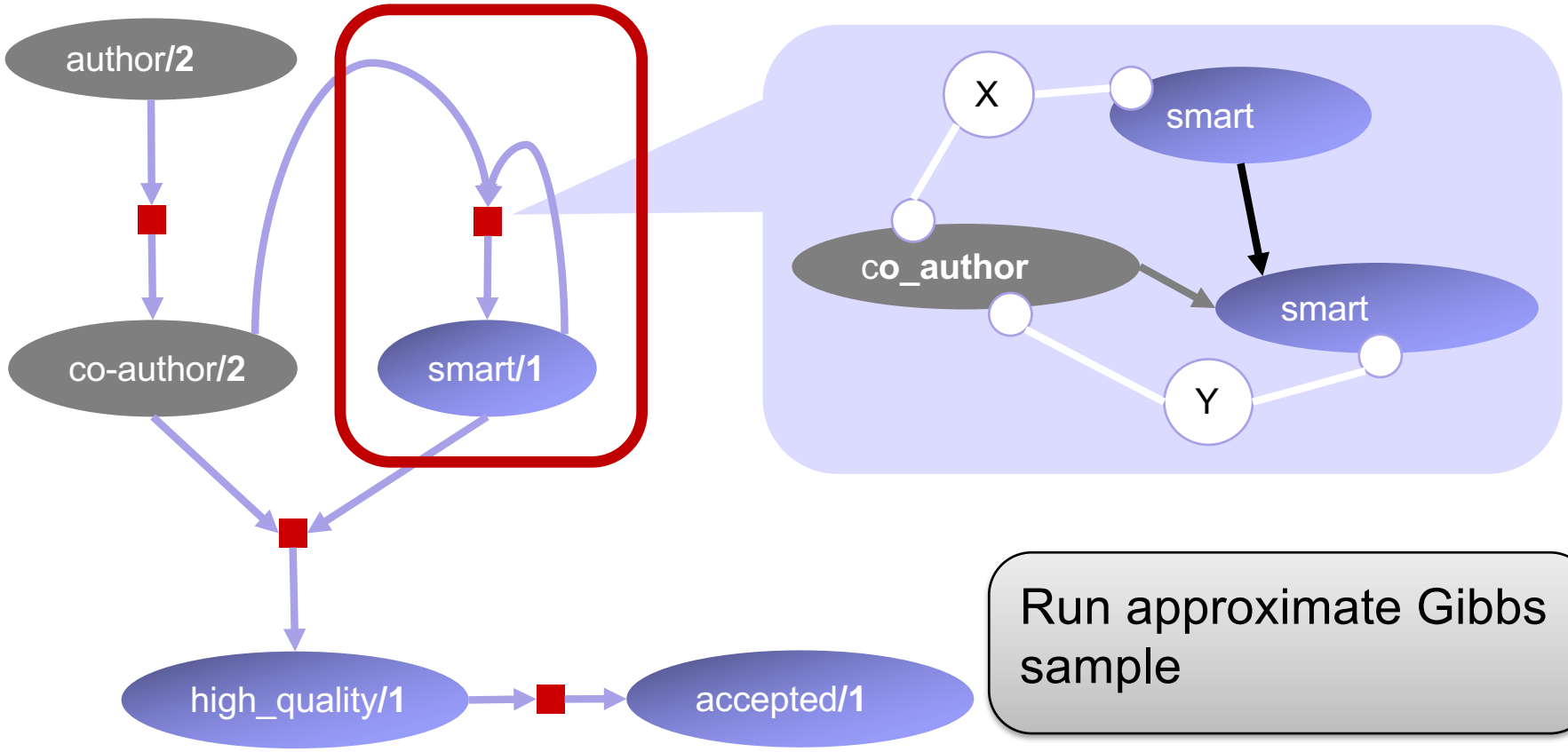
$$\forall x\ author(x,p) \wedge smart(x) \Rightarrow high\_quality(p)$$

$$\forall x\ high\_quality(p) \Rightarrow accepted(p)$$

cyclic dependency

$$\boxed{\forall x,y\ co\_author(x,y) \wedge smart(x) \Rightarrow smart(y)}$$

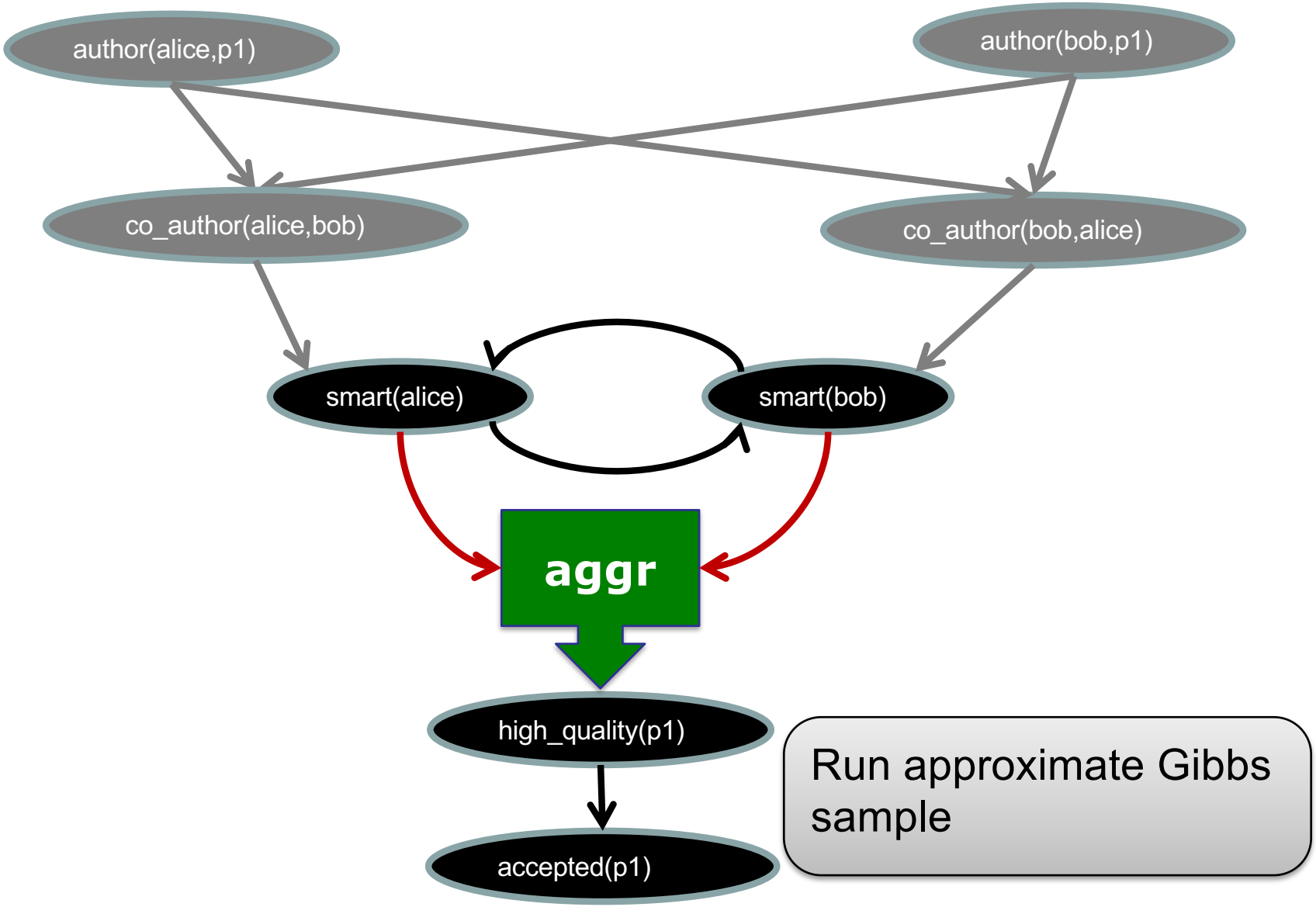
$$\forall x,y\ \exists p\ author(x,p) \wedge author(y,p) \Rightarrow co\_author(x,y)$$



Run approximate Gibbs sample

[Neville, Jensen 2007]

Option 1 : Relational Dependency Networks (RDNs)

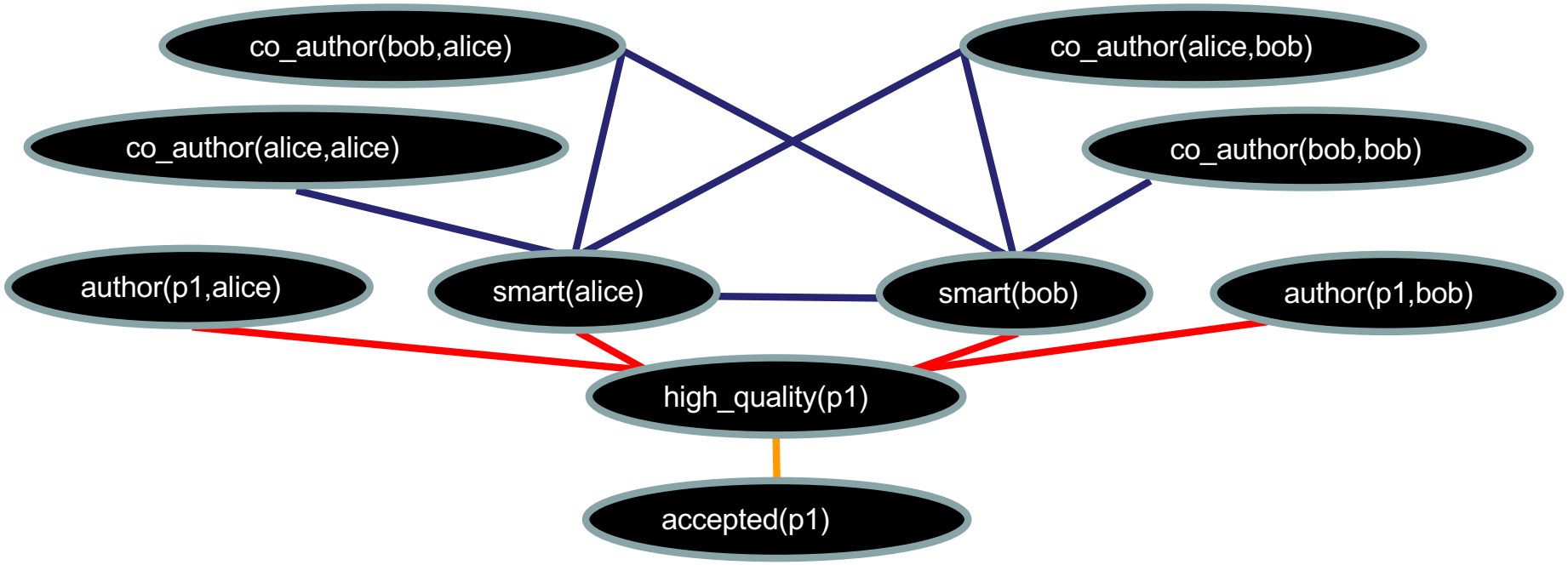


Run approximate Gibbs sample

Option 2: Markov Logic Networks

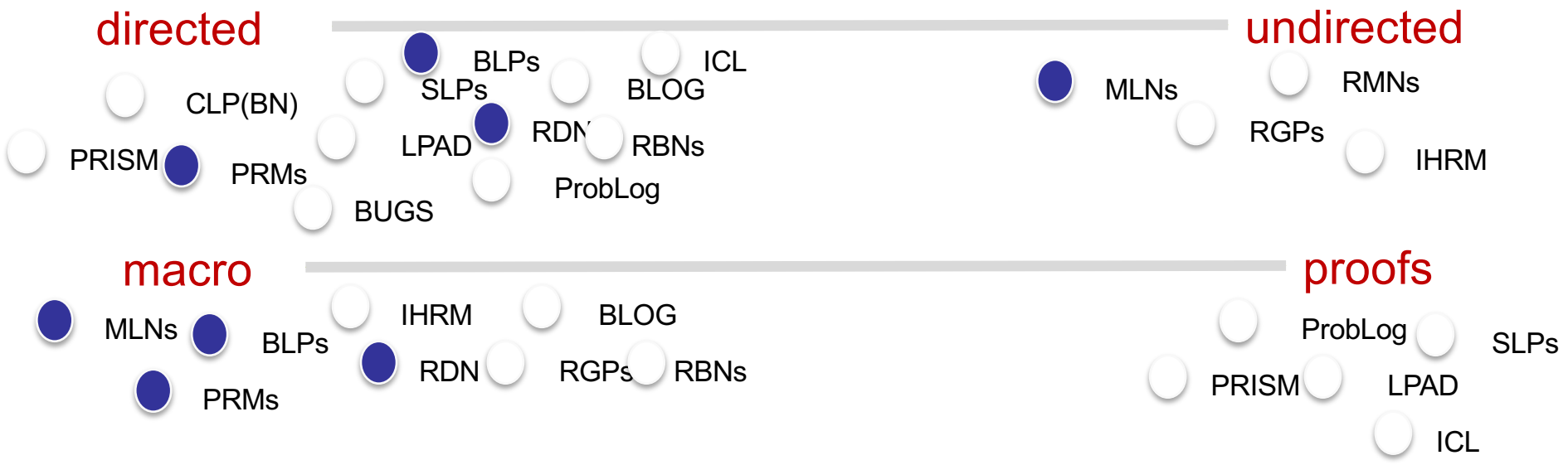
Suppose we have constants: **alice**, **bob** and **p1**

1.5	$\forall x\ author(x,p) \wedge smart(x) \Rightarrow high\_quality(p)$
1.1	$\forall x\ high\_quality(p) \Rightarrow accepted(p)$
1.2	$\forall x,y\ co\_author(x,y) \Rightarrow (smart(x) \Leftrightarrow smart(y))$
∞	$\forall x,y\ \exists p\ author(x,p) \wedge author(y,p) \Rightarrow co\_author(x,y)$

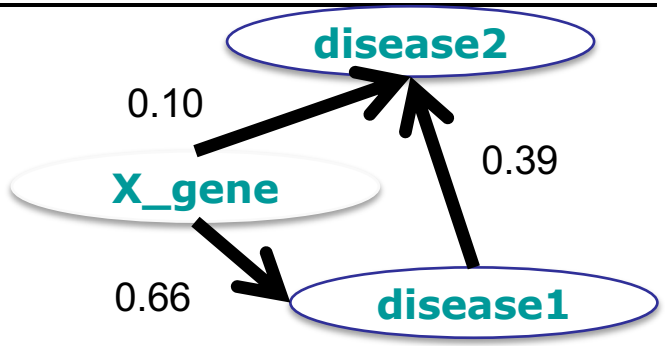


Compile to an undirected model

Key Dimensions with some prototypes



ProbLog



0.10 :: edges(x_gene, disease2)
0.66 :: edge(x_gene, disease1)
0.39 :: edges(disease1,disease2)

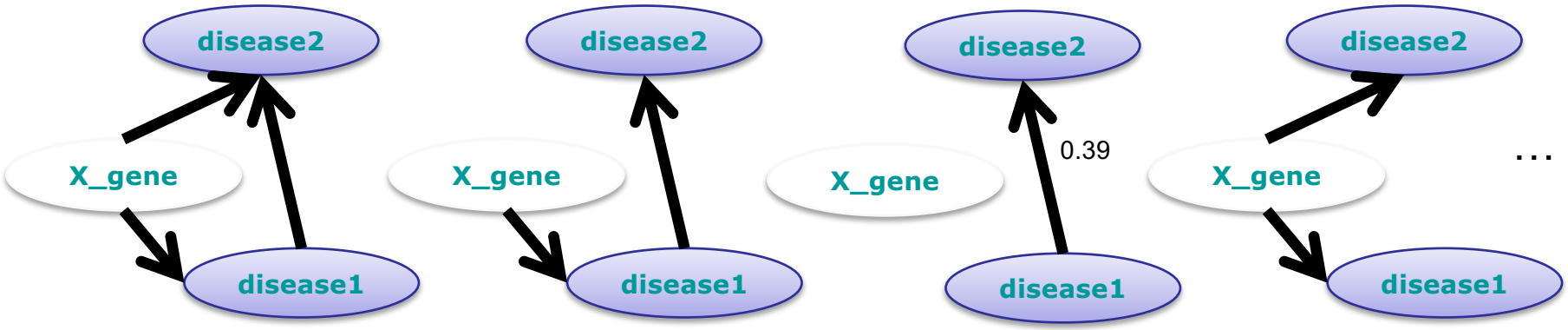
path(X,Y) :- edge(X,Y)
path(X,Y) :- edges(X,Z), path(Z,Y)

Label of a clause/fact c is the probability that c belongs to the target program; Facts/clauses independent of each other

Defines a distribution over programs

P(path(x_gene,disease2))= sum of probs of all programs that entail the query

P=0.1*0.66*0.39 + P=(1-0.1)*0.66*0.39 ... + P=0.1*0.66*(1-0.39)



Exponentially many subprograms! To avoid explosion, consider proofs/paths only + store them e.g. in a BDD in order to count correctly

Many other dimensions!!

TECHNISCHE UNIVERSITÄT DARMSTADT

directed — **undirected**

BLPs, ICL, SLPs, BLOG, CLP(BN), RDN, LPAD, RBNs, PRISM, PRMs, ProbLog, BUGS

MLNs, RMNs, RGPs, IHRM

macro — **proofs**

IHRM, BLOG, MLNs, BLPs, RDN, RGPs, RBNs, PRMs

ProbLog, SLPs, PRISM, LPAD, ICL

parametric — **non-parametric**

ProbLog, BLPs, ICL, LPAD, BLOG, MLNs, SLPs, RDN, PRISM, RBNs, PRMs

IHRM, RGPs, NP-BLOG

CWA — **OWA**

ProbLog, RBNs, BLPs, LPAD, IHRM, ICL, MLNs, SLPs, RDN, PRISM, RGPs, PRMs

NP-BLOG, MEBNs, BLOG

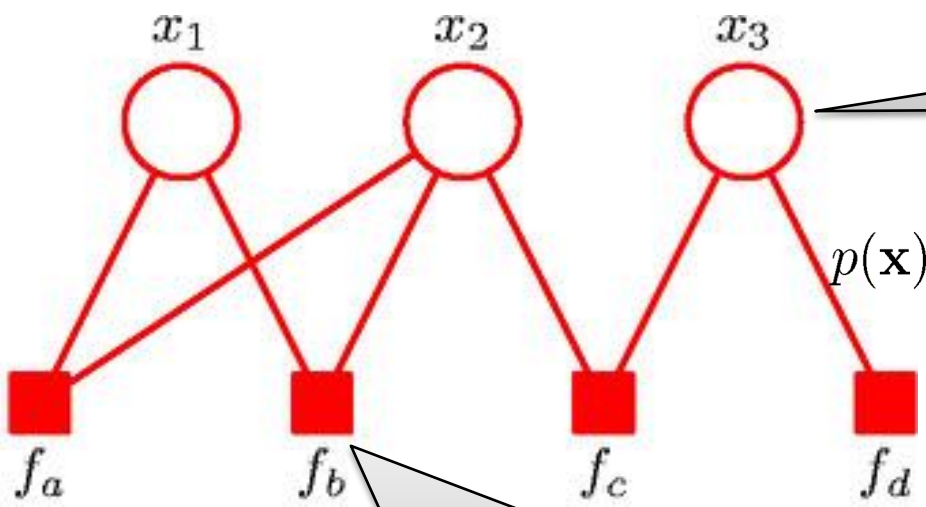
And actually they span the whole AI spectrum

Relational topic models
Mixed-membership models
Relational Gaussian processes
Relational reinforcement learning
(Partially observable) MDPs
Systems of linear equations
Kalman filters
Declarative information networks
…

So, should we worry about the soup?

This soup boiled down to Graphical Models as intermediate representation

Distributions can naturally be represented as Factor Graphs



Random variable

$$p(\mathbf{x}) = f_a(x_1, x_2) f_b(x_1, x_2) f_c(x_2, x_3) f_d(x_3)$$

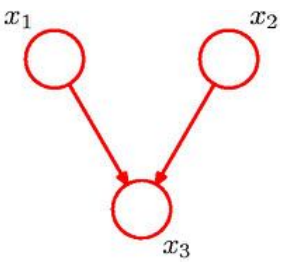
$$p(\mathbf{x}) = \prod_s f_s(\mathbf{x}_s)$$

unnormalized !

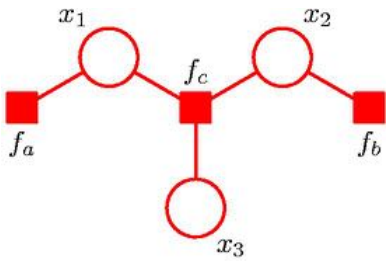
Factor resp. potential

There is an edge between a circle and a box if the variable is in the domain/scope of the factor

Factor Graphs from Graphical Models



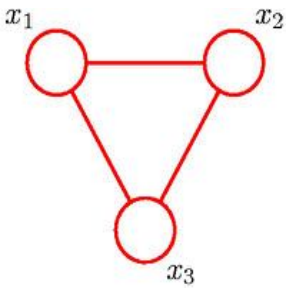
$$p(\mathbf{x}) = p(x_1)p(x_2)\, p(x_3|x_1,x_2)$$



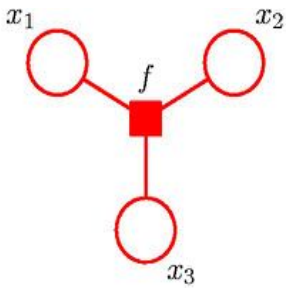
$$f_a(x_1) = p(x_1)$$

$$f_b(x_2) = p(x_2)$$

$$f_c(x_1,x_2,x_3) = p(x_3|x_1,x_2)$$



$$\psi(x_1,x_2,x_3)$$



$$f(x_1,x_2,x_3) = \psi(x_1,x_2,x_3)$$

Similar "boiling down" process is going on in StarAI!

Boiled-Down SRL Alphabet Soup

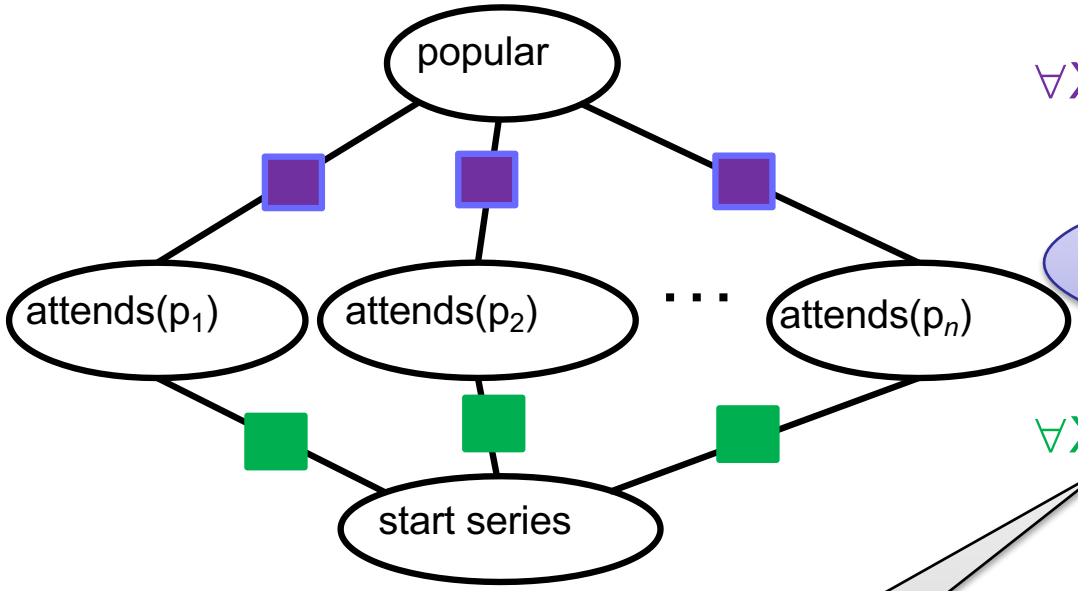
Given a relational model in your language of choice, a set of constants and a query, **compile** everything **into an intermediate respresentation**

- **(logically parameterized) Factor graphs**
- **BDDs, Artihmetic Circuits, d-DNNFs, …**
- **Weighted CNFs**

Run (lifted) **inference**

Rules + Potential: Logically Parameterized Factors (parfactors)

[Poole 2003; de Salvo Braz et al. 2005,…]



$\forall X.\ \phi_1(\text{popular}, \text{attends}(X))$

Logical Variables parameterize RV

$\forall X.\ \phi_2(\text{attends}(X), \text{series})$

Atoms represent a set of random variables

Parfactors parameterized factors

There can also be contraints to logical variables such as X=/=ACAI2018

Rules + Weights: Weighted CNF

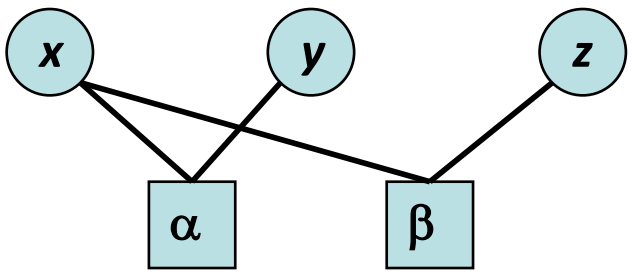
- Weighted MAX-SAT as mode finding for log-linear distributions
- Each configuration has a cost: the sum of the weights of the unsatisfied (ground) clauses.
- An infinite cost gives a ‘hard’ clause.
- Goal: find an assignment with minimal cost.

Weigthed CNF

w1 w2

$$\underbrace{(x \vee y)}_{\alpha} \wedge \underbrace{(\neg x \vee z)}_{\beta}$$

Factor Graph:



x	y	$f_\alpha(x,y)$
0	0	0
0	1	w1
1	0	w1
1	1	w1

x	z	$f_\beta(x,z)$
0	0	w2
0	1	w2
1	0	0
1	1	w2

[Darwiche et al.; Van den Broeck et al. and many others]

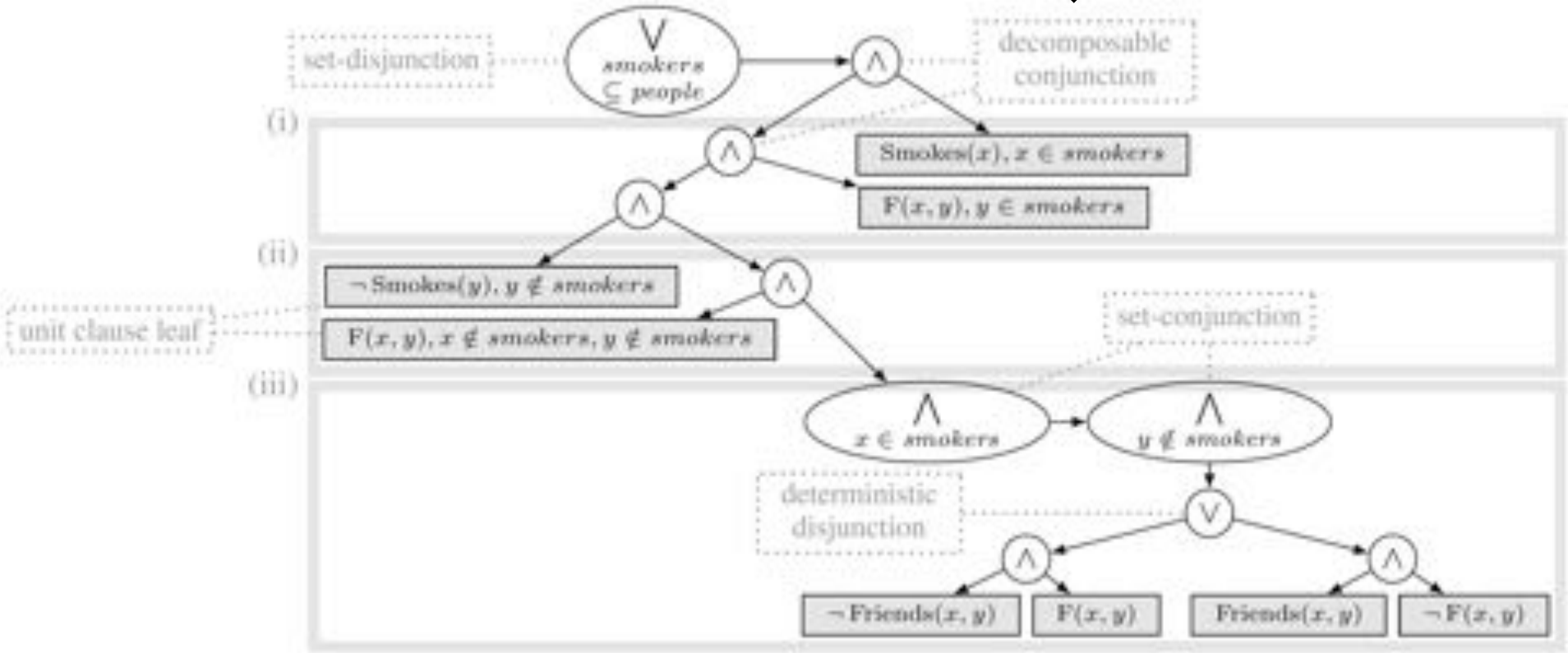
Rules + Weights: Knowledge Compilation

$$w \quad \text{Smokes}(x) \wedge \text{Friends}(x,y) \Rightarrow \text{Smokes}(y).$$

$$w \quad \text{F}(x,y)$$

$$\infty \quad \text{F}(x,y) \equiv [\text{Smokes}(x) \wedge \text{Friends}(x,y) \Rightarrow \text{Smokes}(y)]\,.$$

d-DNNF



And even low-level C/C++ programs

$$\{\langle\{x,m\}, r(x,m) \wedge s(x,m), 1.2\rangle, \langle\{x,m\}, s(x,m) \wedge t(x), 0.2\rangle\}$$

```
v2=0;
for (int i = 0; i <= 2; i++){
    v5 = 0;
    for (int j = 0; j <= i; j++)
        v5 += choose(i, j) * exp(1.2 * j);
    v2 += choose(2, i) * pow(2, 2-i)
            * v5 * (exp(0.2 * i) + 1);
}
v1 = pow(v2, 4);
```

What have we learnt so far in Part II?

- There are several ways to specify relational probabilistic models
- The goal is not to have a probabilistic characterization
- Exising Frameowrks highlight different aspects of relational modeling
- Semantically this soup boiled down to weighted CNFs, factor graphs, parfactors, diagrams and even program code

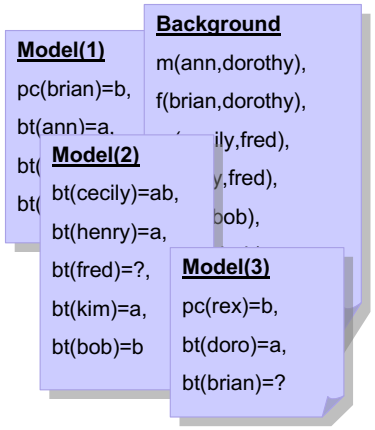
How do we learn relational models?

Goals

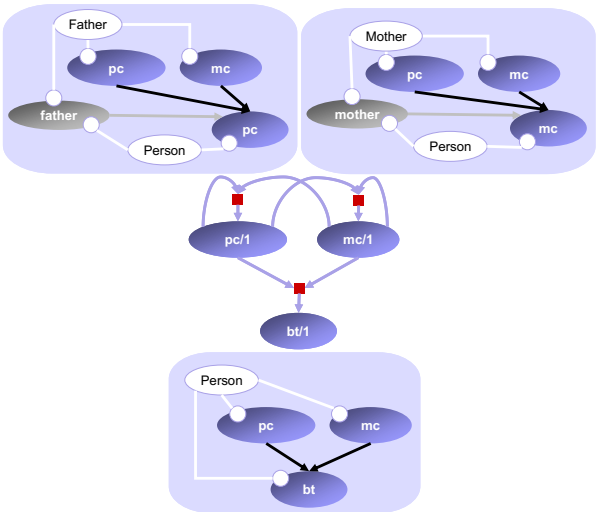
- Parameter Estimation
- (Vanilla) relational learning approach
- nFOIL, Hypergraph Lifting, and Boosting

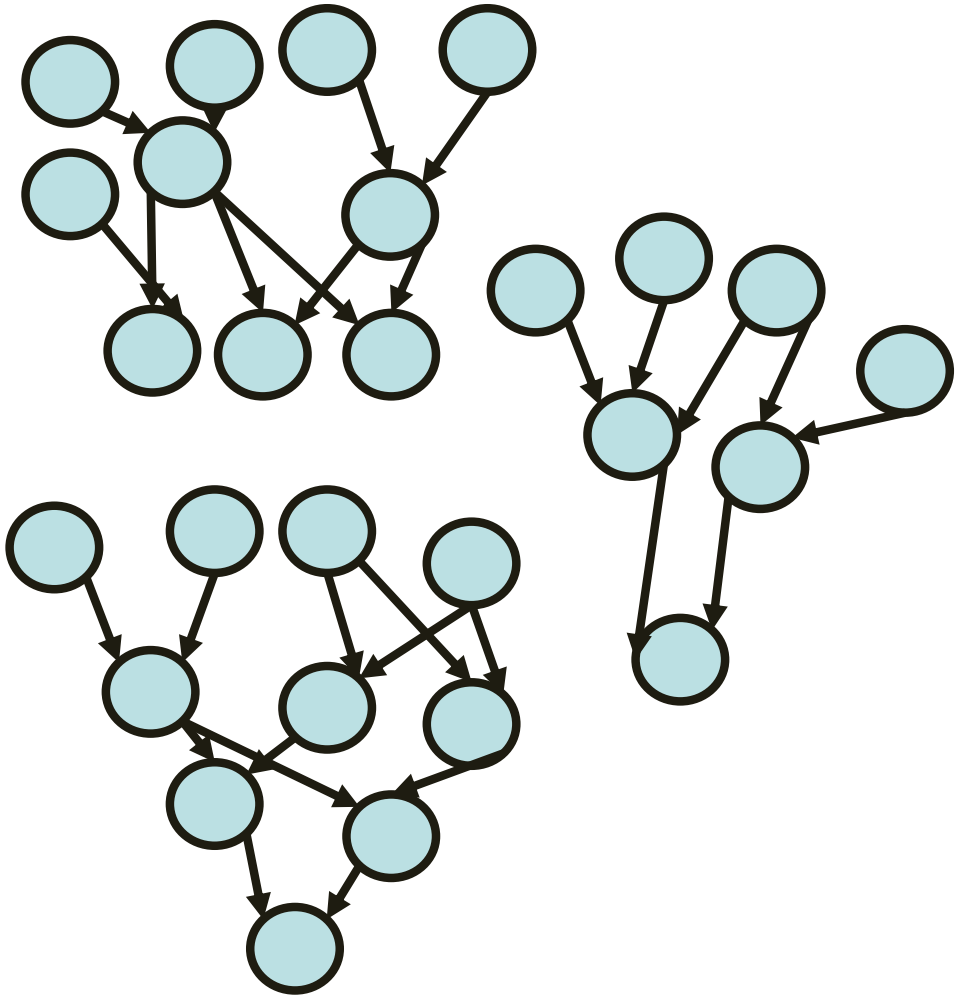
PARAMETER ESTIMATION FOR RELATIONAL MODELS

Relational Parameter Estimation

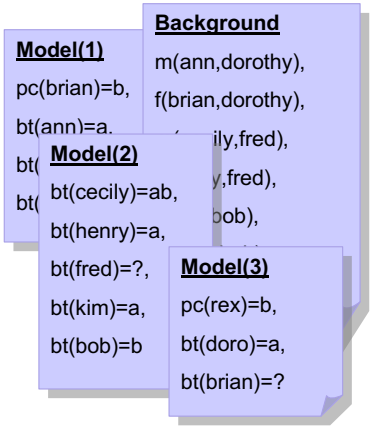


+

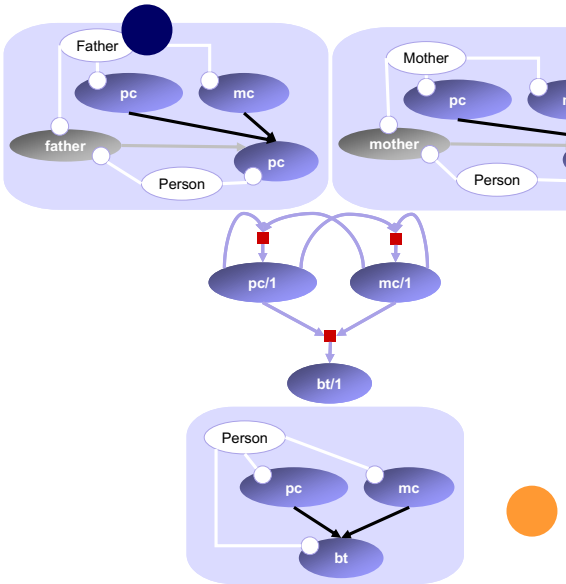


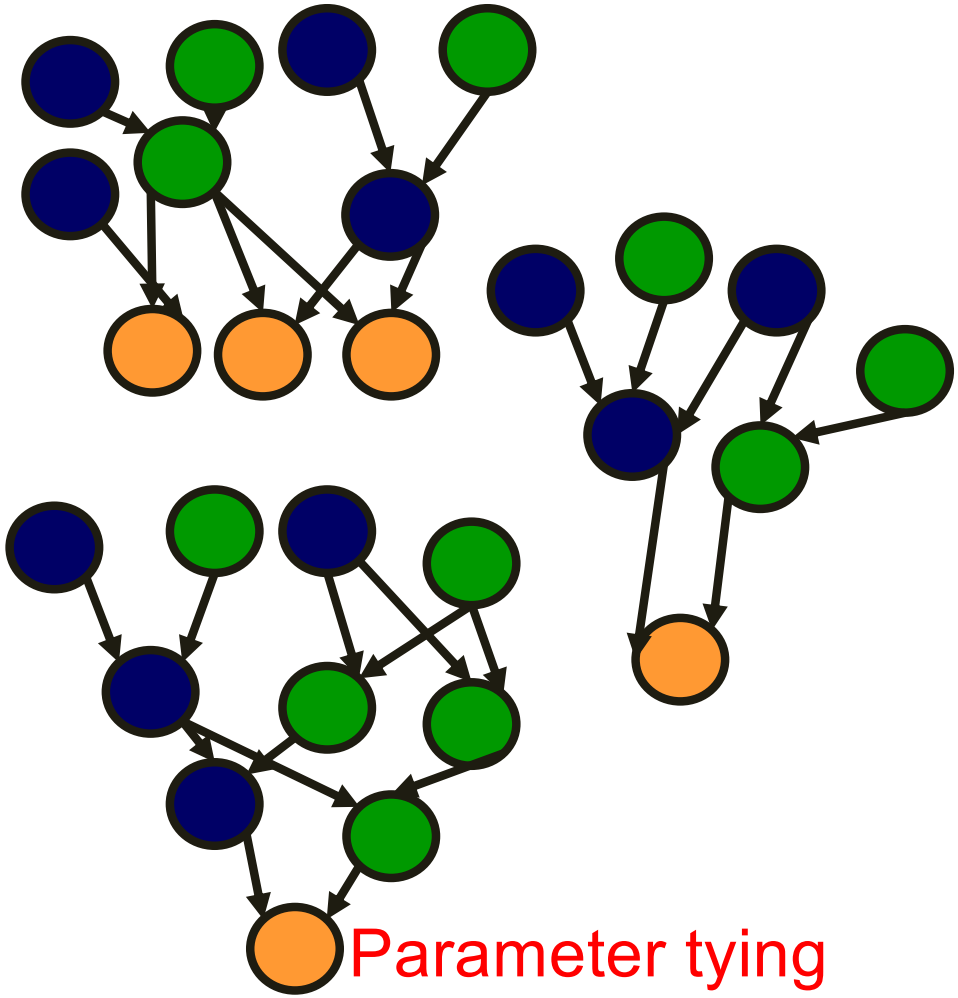


Relational Parameter Estimation



+

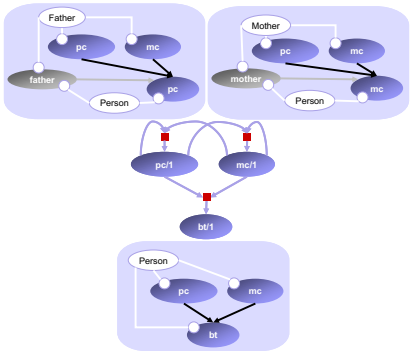




So, we can apply „standard“ EM

TECHNISCHE UNIVERSITÄT DARMSTADT

Logic Program L



Initial Parameters q0

iterate until convergence

Expectation

Model(1)
pc(brian)=b,
bt(ann)=a.
bt(
bt(

Background
m(ann,dorothy),
f(brian,dorothy),
ily,fred),
y,fred),
bob),

Model(2)
bt(cecily)=ab,
bt(henry)=a,
bt(fred)=?,
bt(kim)=a,
bt(bob)=b

Model(3)
pc(rex)=b,
bt(doro)=a,
bt(brian)=?

Inference

Current Model (M,qk)

Expected counts of a clause

$$\sum_{\text{Ground Instance GI}} \sum_{\text{DataCase DC}} \mathbf{P(\ head(GI),\ body(GI)\ |\ DC\ )}$$

$$\frac{\sum_{\text{Ground Instance GI}} \sum_{\text{DataCase DC}} \mathbf{P(\ head(GI),\ body(GI)\ |\ DC\ )}}{\sum_{\text{Ground Instance GI}} \sum_{\text{DataCase DC}} \mathbf{P(\ body(GI)\ |\ DC\ )}}$$

Maximization

Update parameters (ML, MAP)

Generative Learning

Function to optimize:

$$f(w) = \sum_i w_i n_i(x) - \log Z$$

$$Z = \sum_x \exp\left(\sum_i w_i n_i(x')\right)$$

Gradient:

$$\frac{\partial}{\partial w_j} f(w) = n_j(x) - \frac{1}{Z}\sum_{x'} \exp\left(\sum_i w_i n_i(x')\right) n_j(x')$$

$$= n_j(x) - \sum_{x'} P(x') n_j(x')$$

$$= \underbrace{n_j(x)}_{} - \underbrace{E[n_j(x)]}_{}$$

Counts in training data

Weighted sum over all possible worlds
No evidence, just sets of constants
Very hard to approximate

It is #P-complete to count the number of true groundings.
Therefore, one often sticks to approximations

Pseudo-likelihood

Function to optimize:

$$PL(x) = \prod_l P(X_l = x_l \mid MB(x_l))$$

$$\log PL(x) = \sum_l \log P(X_l = x_l \mid MB(x_l))$$

$$P(X_l = x_l \mid MB(x_l)) = \frac{P(x)}{P(x_{[X_l=0]}) + P(x_{[X_l=1]})}$$

$$= \frac{1/Z \exp(\Sigma w_i n_i(x))}{1/Z \exp(\Sigma w_i n_i(x_{[X_l=0]})) + 1/Z \exp(\Sigma w_i n_i(x_{[X_l=1]}))}$$

Gradient:

$$\frac{\partial}{\partial w_j} \log PL(x) = \sum_l n_j(x) - P(X_l = 0 \mid MB(X_l))\, n_j(x_{[X_l=0]}) - P(X_l = 1 \mid MB(X_l))\, n_j(x_{[X_l=1]})$$

$$= \sum_l n_j(x) - E_{x'_l}[n_j(x_{[X_l=x'_l]})]$$

Pseudo-likelihood

Function to optimize:

$$PL(x) = \prod_l P(X_l = x_l \mid MB(x_l))$$

$$\log PL(x) = \sum_l \log P(X_l = x_l \mid MB(x_l))$$

$$P(X_l = x_l \mid MB(x_l)) = \frac{P(x)}{P(x_{[X_l=0]}) + P(x_{[X_l=1]})}$$

While effective, still hard to count in many data sets

Approximate counting techniques exist
(Sarkhel et al. AAAI 2016, Das et al. SDM 2016)

Gr

$$\partial w_j \qquad \sum_l \qquad [X_l=1]))$$

$$- P(X_l = 1 \mid MB(X_l))\, n_j(x_{[X_l=1]})$$

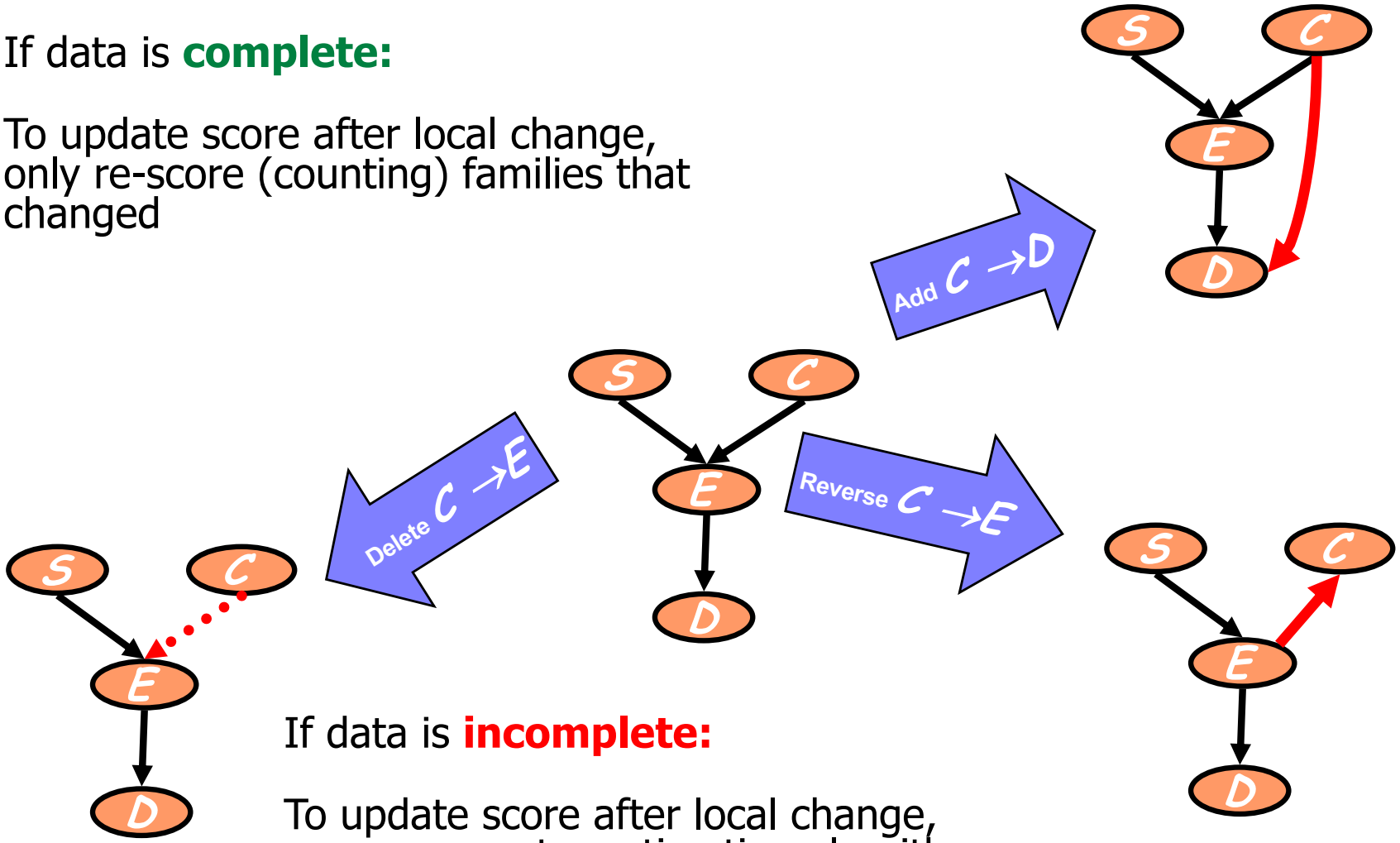
$$= \sum_l n_j(x) - E_{x'_l}[n_j(x_{[X_l=x'_l]})]$$

STRUCTURE LEARNING FOR RELATIONAL MODELS

Probabilistic Graphical Models

If data is **complete:**

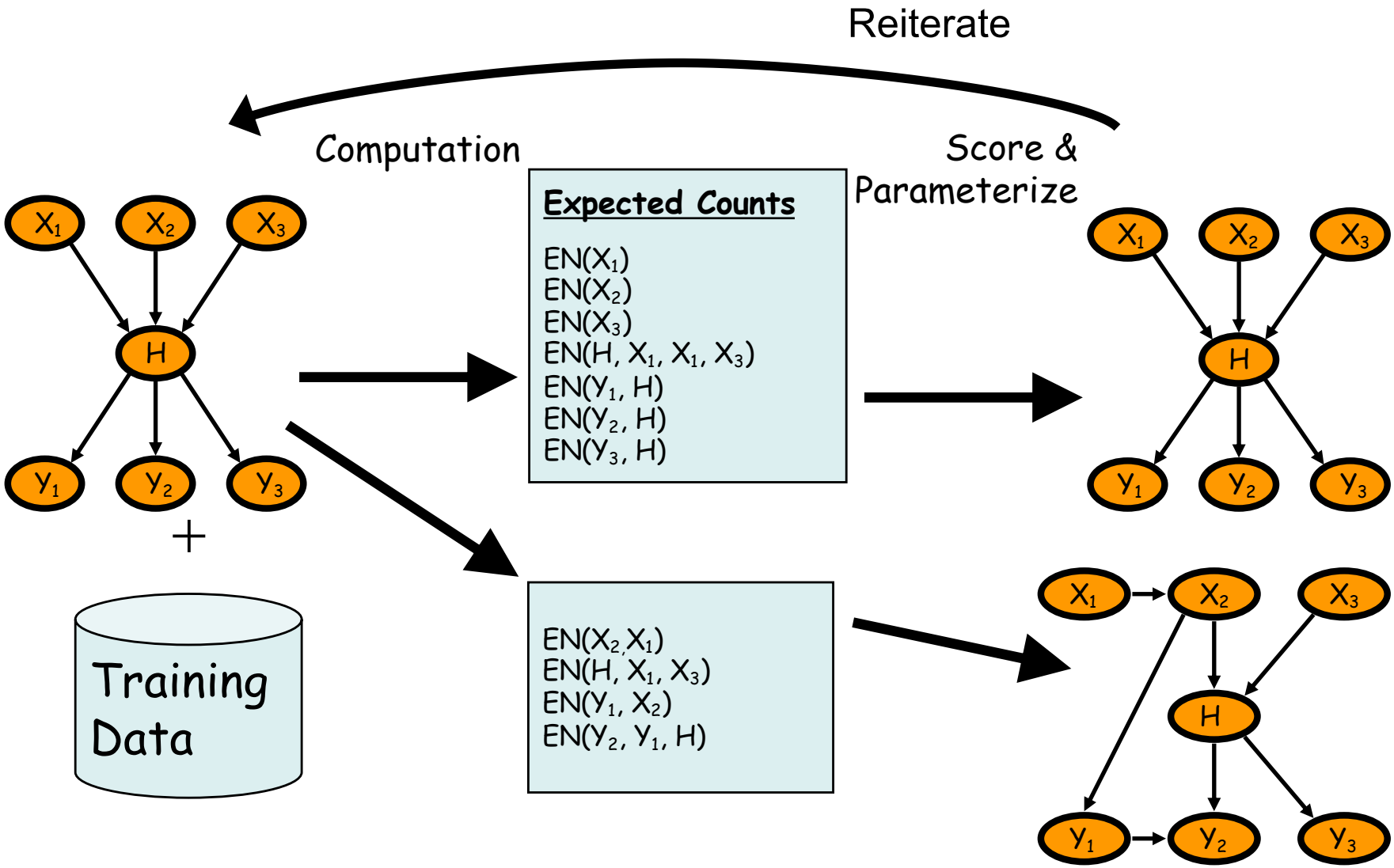
To update score after local change, only re-score (counting) families that changed



If data is **incomplete:**

To update score after local change, rerun parameter estimation algorithm

Structural EM [Friedman et al. 98]



Inductive Logic Programming (ILP) = Machine Learning + Logic Programming

The Problem Specification

[Muggleton, De Raedt JLP96]

Given:

- *Examples:* first-order atomic formulas (atoms), each labeled positive or negative.
- *Background knowledge:* definite clause (if-then rules) theory.
- *Language bias:* constraints on the form of interesting new rules (clauses).

ILP Specification (Continued)

Find:

A hypothesis *h* that meets the language constraints and that, when conjoined with *B*, implies (lets us prove) all of the positive examples but none of the negative examples.

To handle real-world issues such as noise, we often relax the requirements, so that *h* need only entail significantly more positive examples than negative examples.

ILP Specification (illustrated)

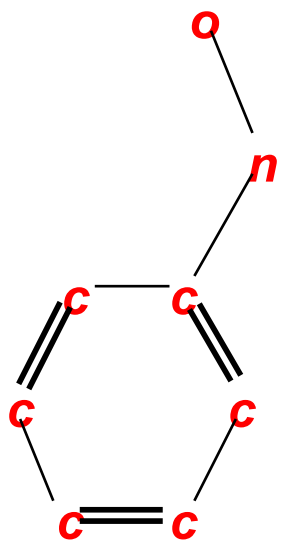
TECHNISCHE UNIVERSITÄT DARMSTADT

Find set of general rules

mutagenic(X) :- atom(X,A,c),charge(X,A,0.82)
mutagenic(X) :- atom(X,A,n),...

Examples E

Pos(mutagenic(m1)
Pos(mutagenic(m2)
Neg(mutagenic(m3)



Background Knowledge B

molecule(m1) molecule(m2)
atom(m1,a11,c) atom(m2,a21,o)
atom(m1,a12,n) atom(m2,a22,n)
bond(m1,a11,a12) bond(m2,a21,a22)
....

A Common Approach

Use a greedy covering algorithm.

Repeat while some positive examples remain uncovered (not entailed):

1. Find a *good clause* (one that covers as many positive examples as possible but no/few negatives).
2. Add that clause to the current theory, and remove the positive examples that it covers.

ILP algorithms use this approach but vary in their method for finding a *good clause*.

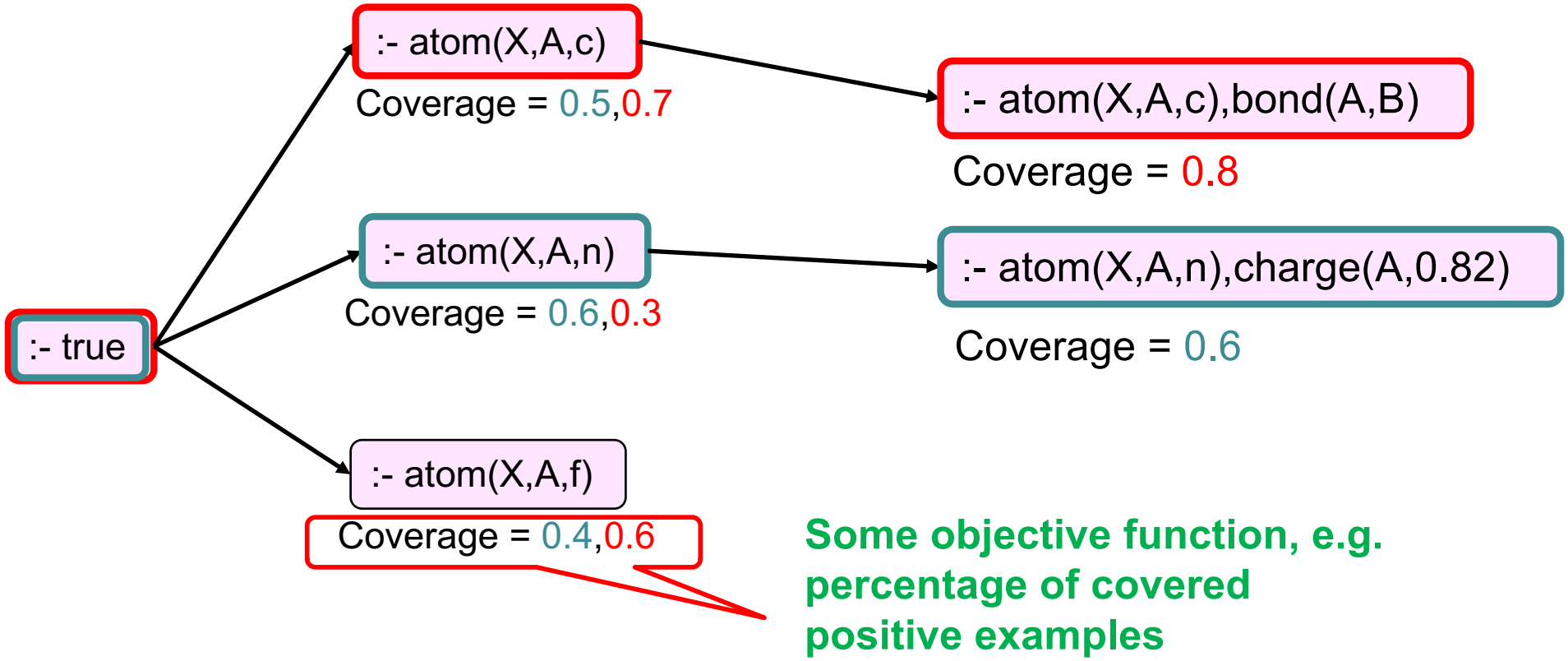
Example ILP Algorithm: FOIL

[Quinlan MLJ 5:239-266, 1990]

mutagenic(X) :- atom(X,A,n),charge(A,0.82)

mutagenic(X) :- atom(X,A,c),bond(A,B)

...



Some objective function, e.g. percentage of covered positive examples

VANILLA STRUCTURE LEARNING FOR PROBABILISTIC RELATIONAL MODELS

Vanilla SRL Approach [De Raedt, Kersting ALT04]

TECHNISCHE UNIVERSITÄT DARMSTADT

mutagenic(X) :- atom(X,A,n),charge(A,0.82)

mutagenic(X) :- atom(X,A,c),bond(A,B)

=0.882

…

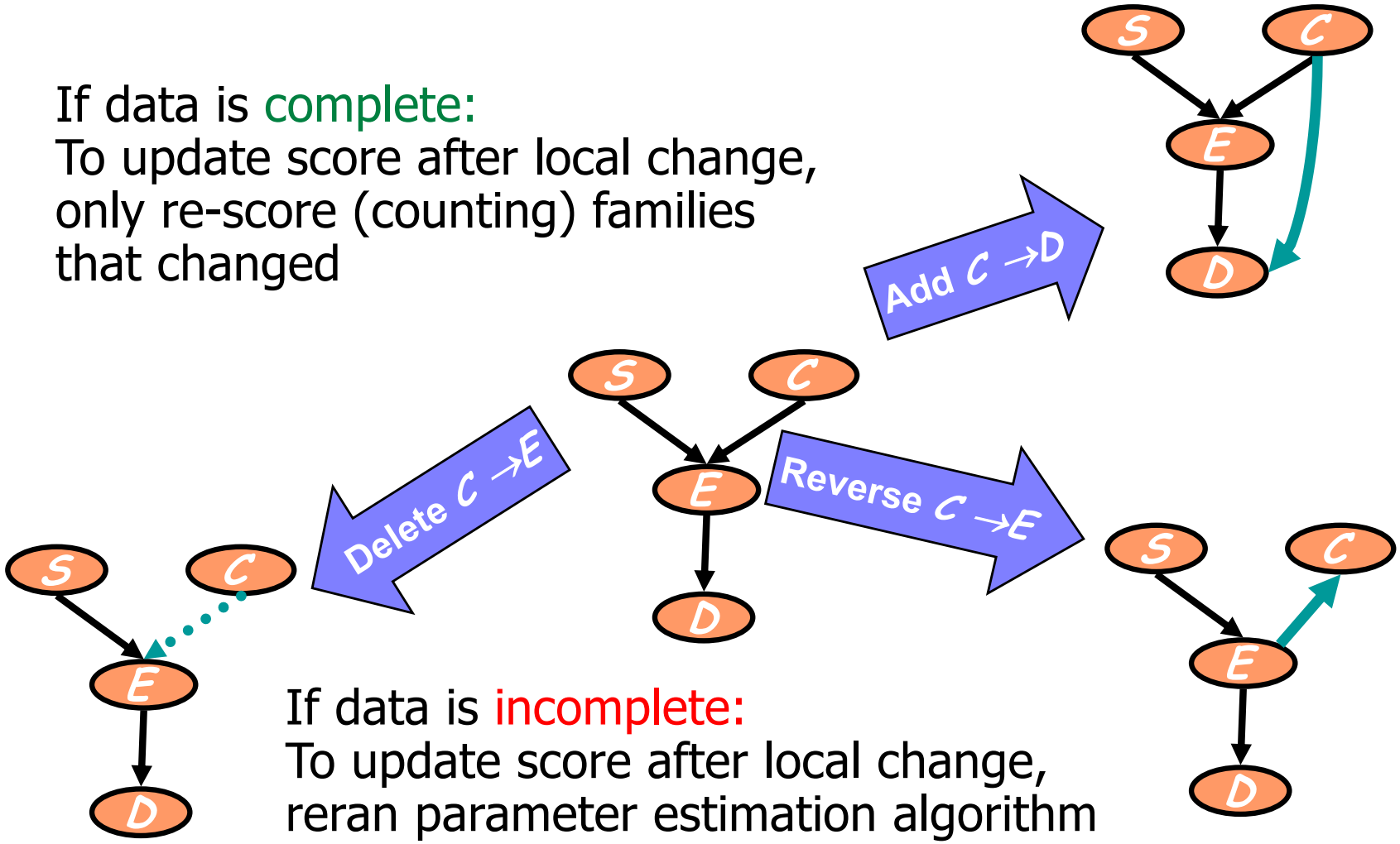
Traverses the hypotheses space a la ILP

Replaces ILP's 0-1 covers relation by a "smooth", probabilistic one [0,1]

$$\begin{aligned} \text{cover}(e, H, B) &= P(e|H, B) \\ \text{cover}(E, H, B) &= \prod_{e \in E} \text{cover}(e, H, B) \end{aligned}$$

So, essentially like in the propositional case!

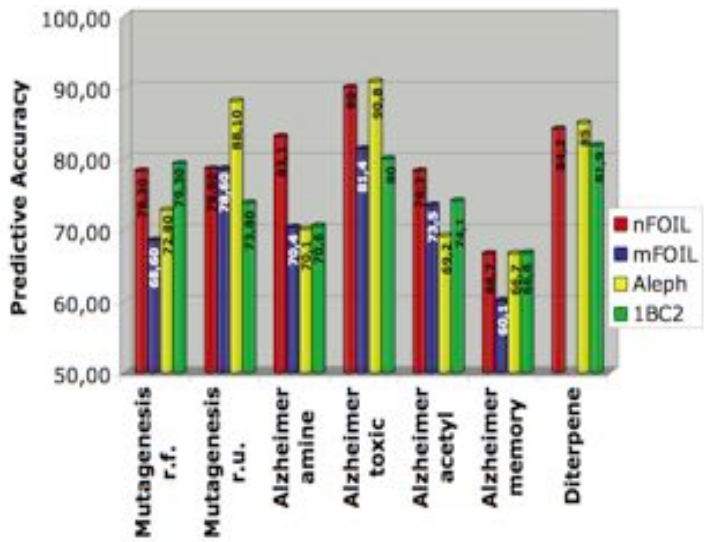
If data is complete:
To update score after local change, only re-score (counting) families that changed



If data is incomplete:
To update score after local change, reran parameter estimation algorithm

nFOIL = FOIL + Naive Bayes

[Landwehr, Kersting, De Raedt JMLR 8(Mar):481-507, 2007]



Clauses are independent features

Likelihood for parameter estimation

Conditional likelihood for scoring clauses

atom(X,A,n),charge(A,0.82)

atom(X,A,c),bond(A,B)

...

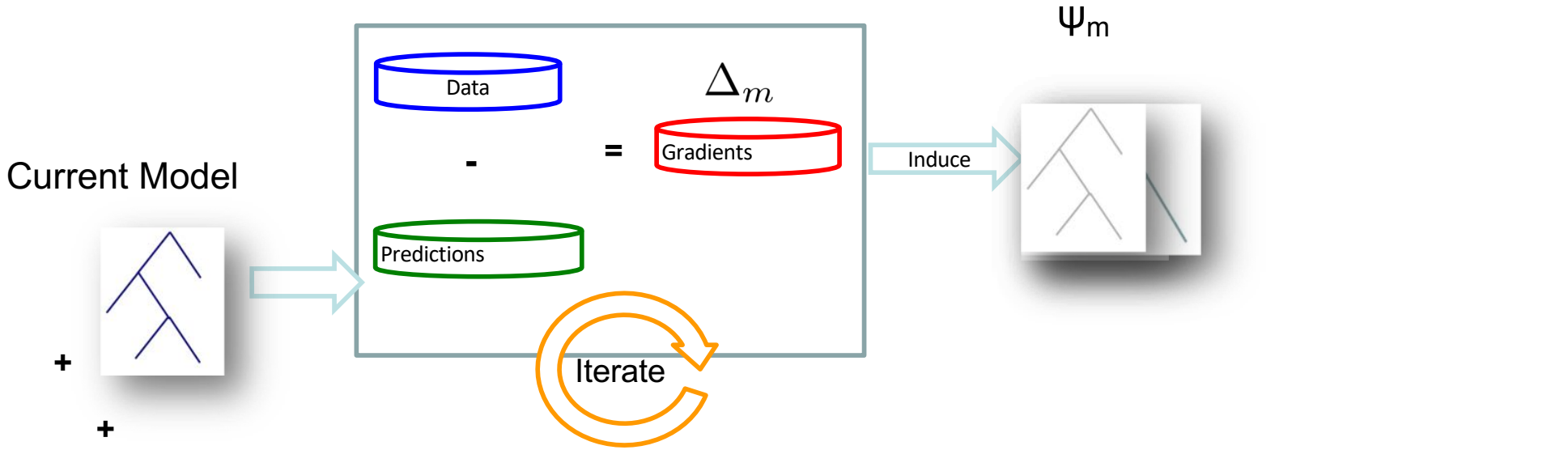
mutagenic(X)

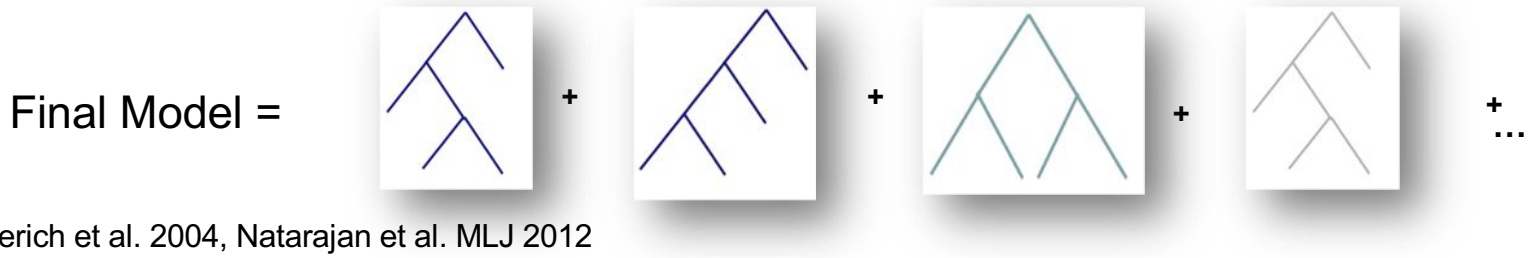
P(truth value clauses**|**truth value target predicate**) x P(**truth value target predicate**)**

STRUCTURE LEARNING MARKOV LOGIC NETWORKS

Ensemble Statistical Relational Learning, here Functional Gradient Boosting

Learn multiple weak models rather than a single complex model





Friedman et al 2001, Dietterich et al. 2004, Natarajan et al. MLJ 2012

Functional Gradients for SRL Models

- Probability of an example

$$P(x_i = true|\mathbf{Pa}(x_i)) \quad = \quad \frac{e^{\psi(x_i;\mathbf{Pa}(x_i))}}{e^{\psi(x_i;\mathbf{Pa}(x_i))} + 1}$$

x	Δ
target(x1)	0.7
target(x2)	-0.2
target(x3)	-0.9

- Functional gradient
  - Maximize

$$LL(\mathbf{X} = \mathbf{x}) \quad = \quad \sum_{x_i \in \mathbf{x}} \log P(x_i|\mathbf{Pa}(x_i))$$

  - Gradient of log-likelihood w.r.t ψ

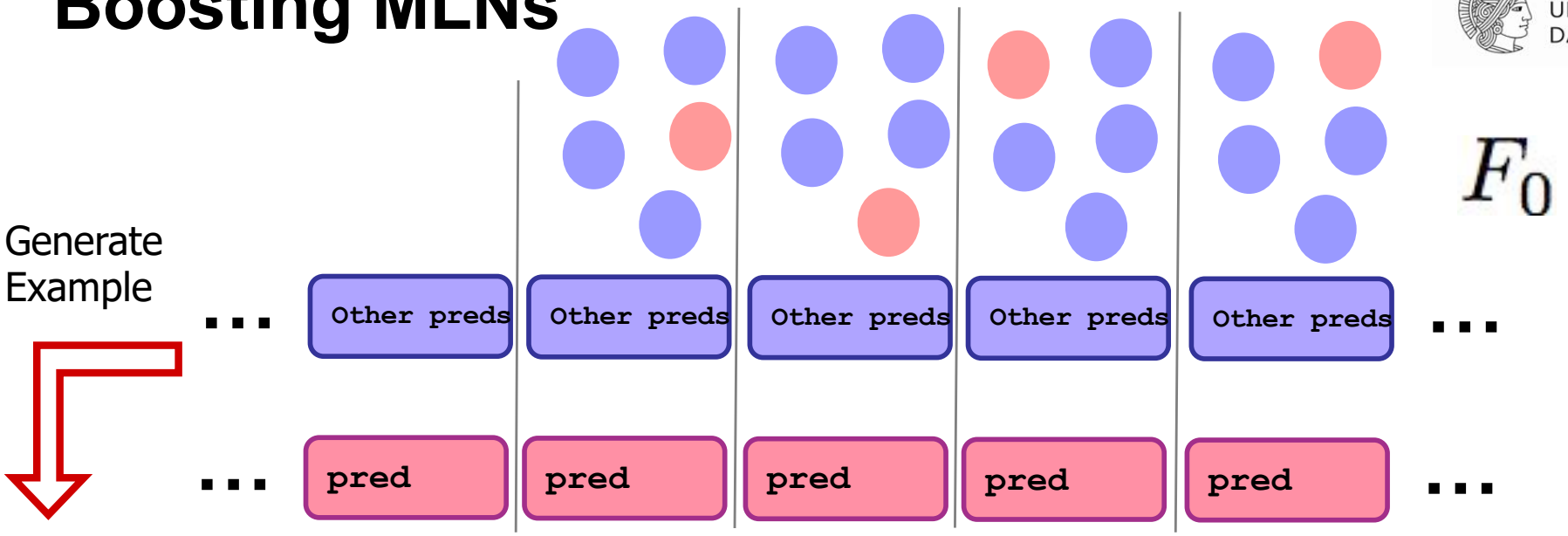
$$\Delta(x_i) = \frac{\partial \log P(\mathbf{X} = \mathbf{x})}{\partial \psi(x_i; \mathbf{Pa}(x_i))} = I(x_i = true; \mathbf{Pa}(x_i)) - P(x_i = true; \mathbf{Pa}(x_i))$$

  - Sum all gradients to get final ψ

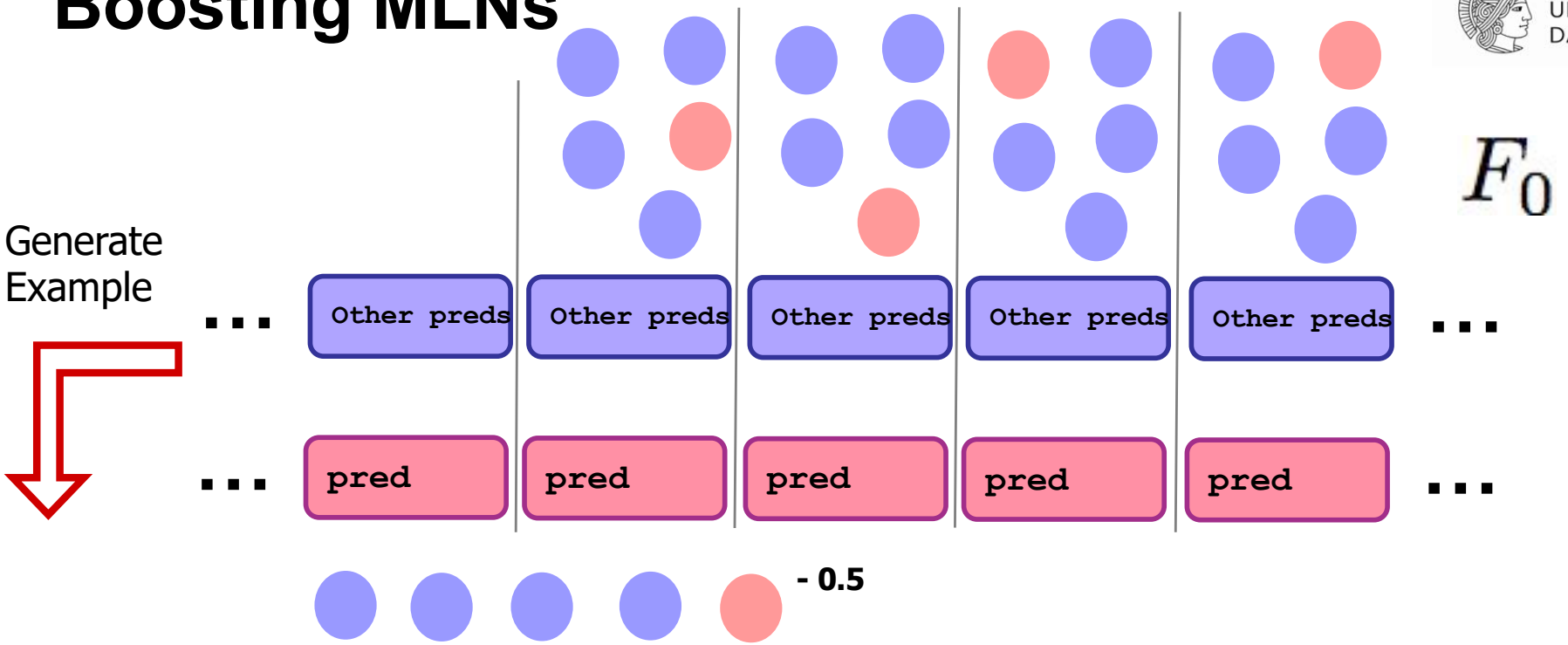
$$\psi_m = \psi_0 + \Delta_1 + ... + \Delta_m$$

Can be extended to multiple SRL models & in presence of hidden data

Boosting MLNs



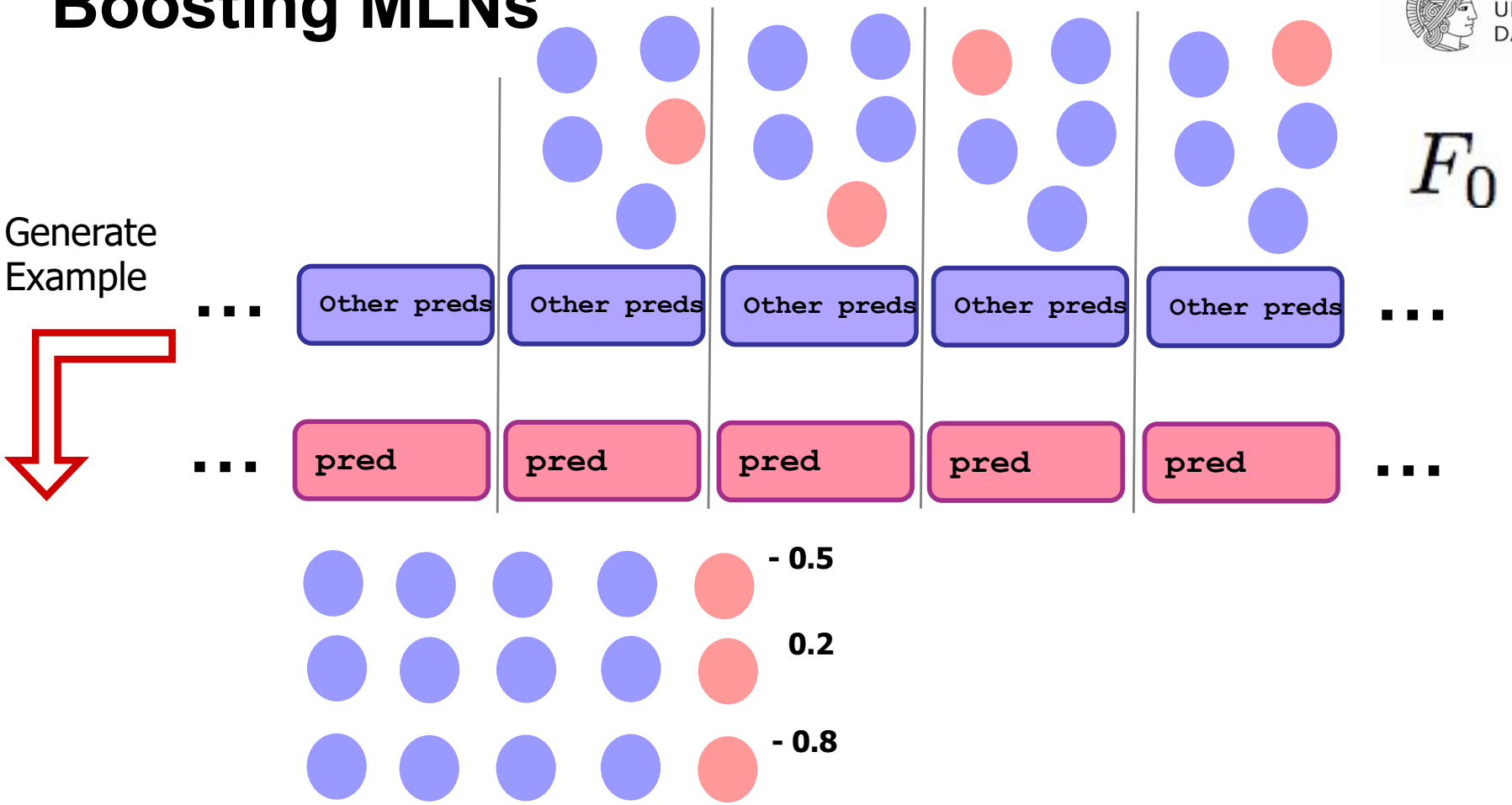
Boosting MLNs



$$logP(y_i|\mathbf{x_i}) = \psi(y_i;\mathbf{x_i}) - log\sum_{y'} e^{\psi(y';\mathbf{x_i})}$$

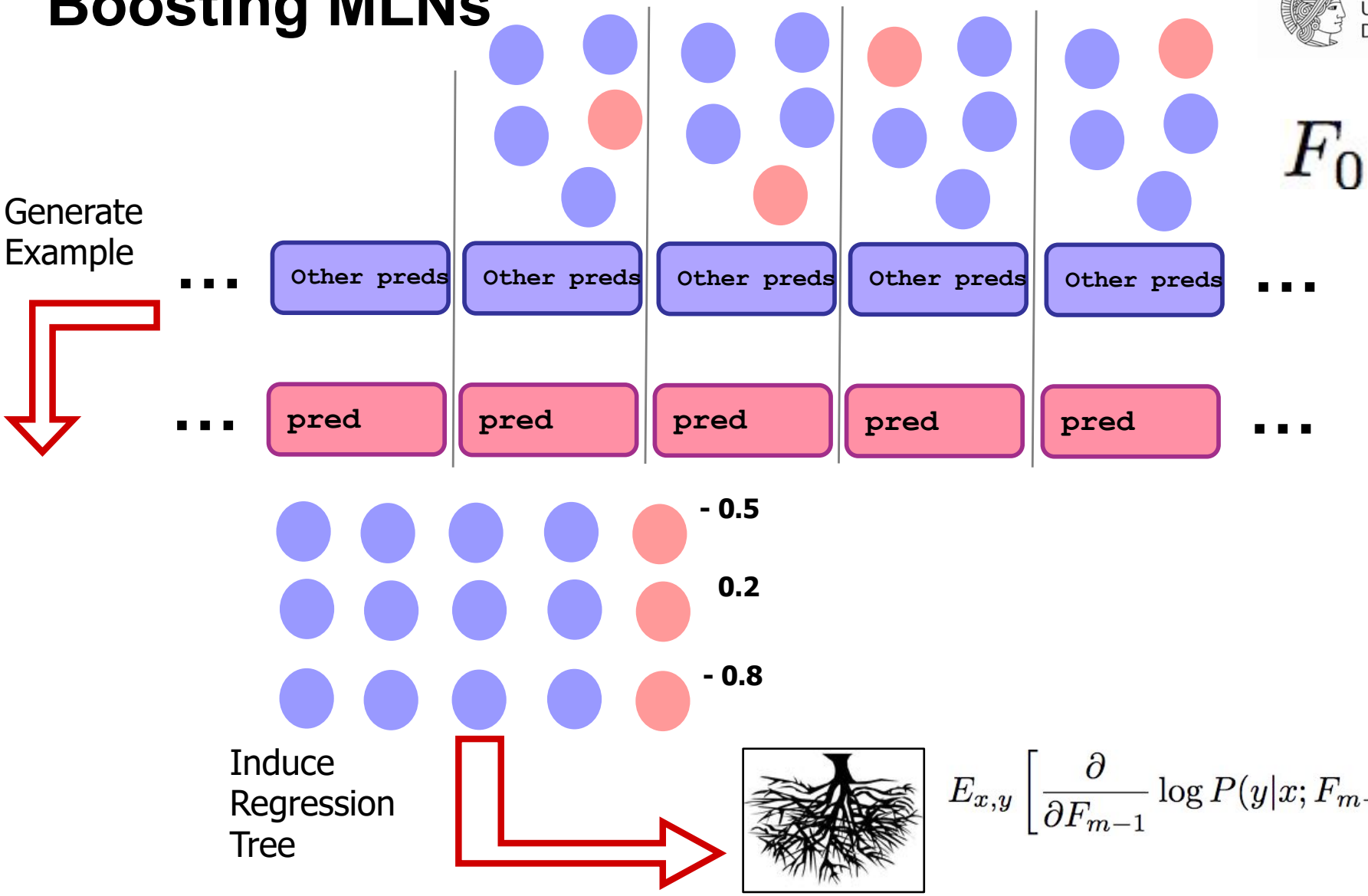
$$\frac{\partial logP(y_i|\mathbf{x_i})}{\partial\psi(y_i=1|\mathbf{x_i})} = I(y_i = 1; x_i) - \frac{e^{\psi(y_i=1;x_i)}}{\sum_{y'} e^{\psi(y';x_i)}}$$

Boosting MLNs



F_0

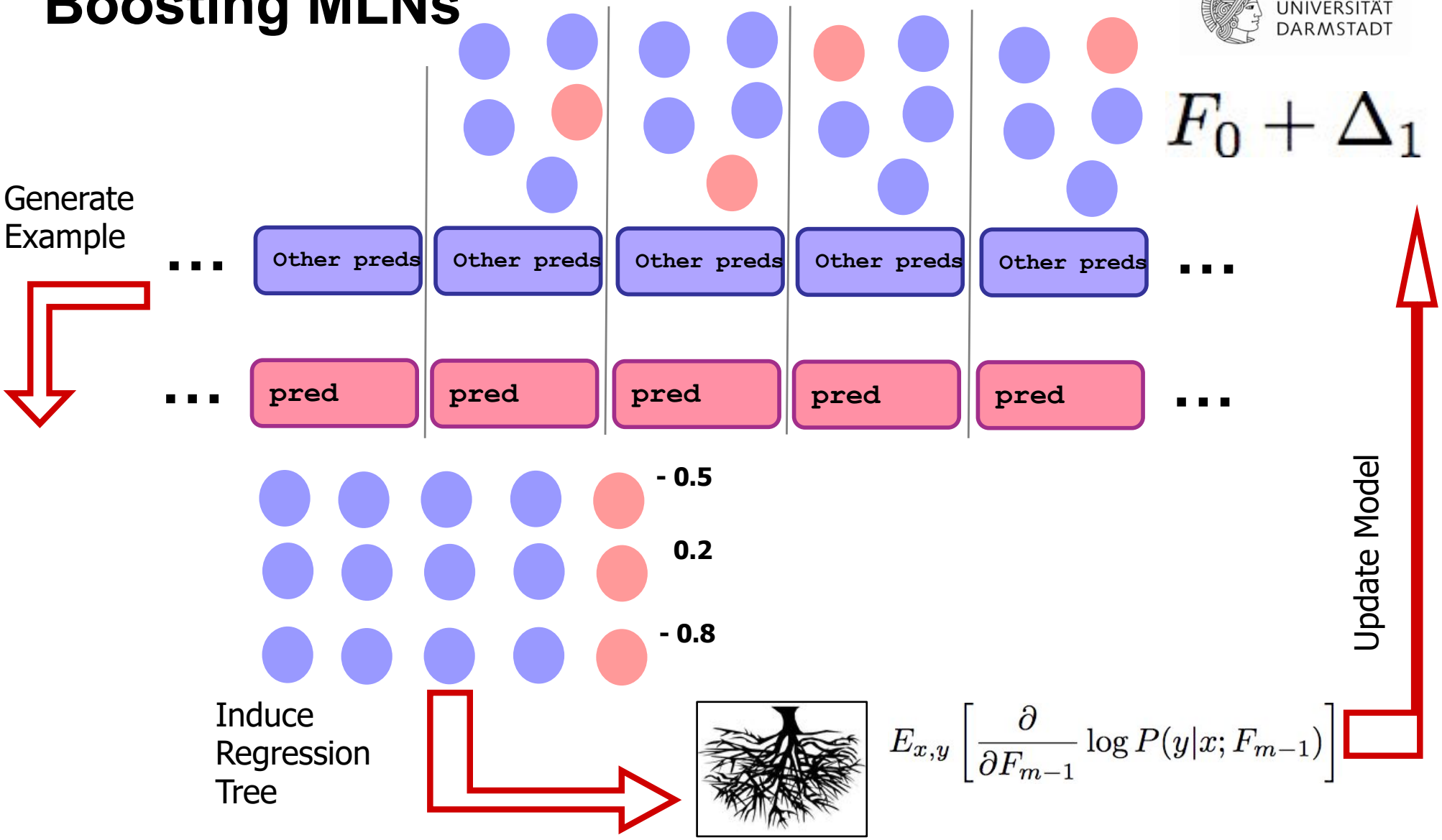
Boosting MLNs



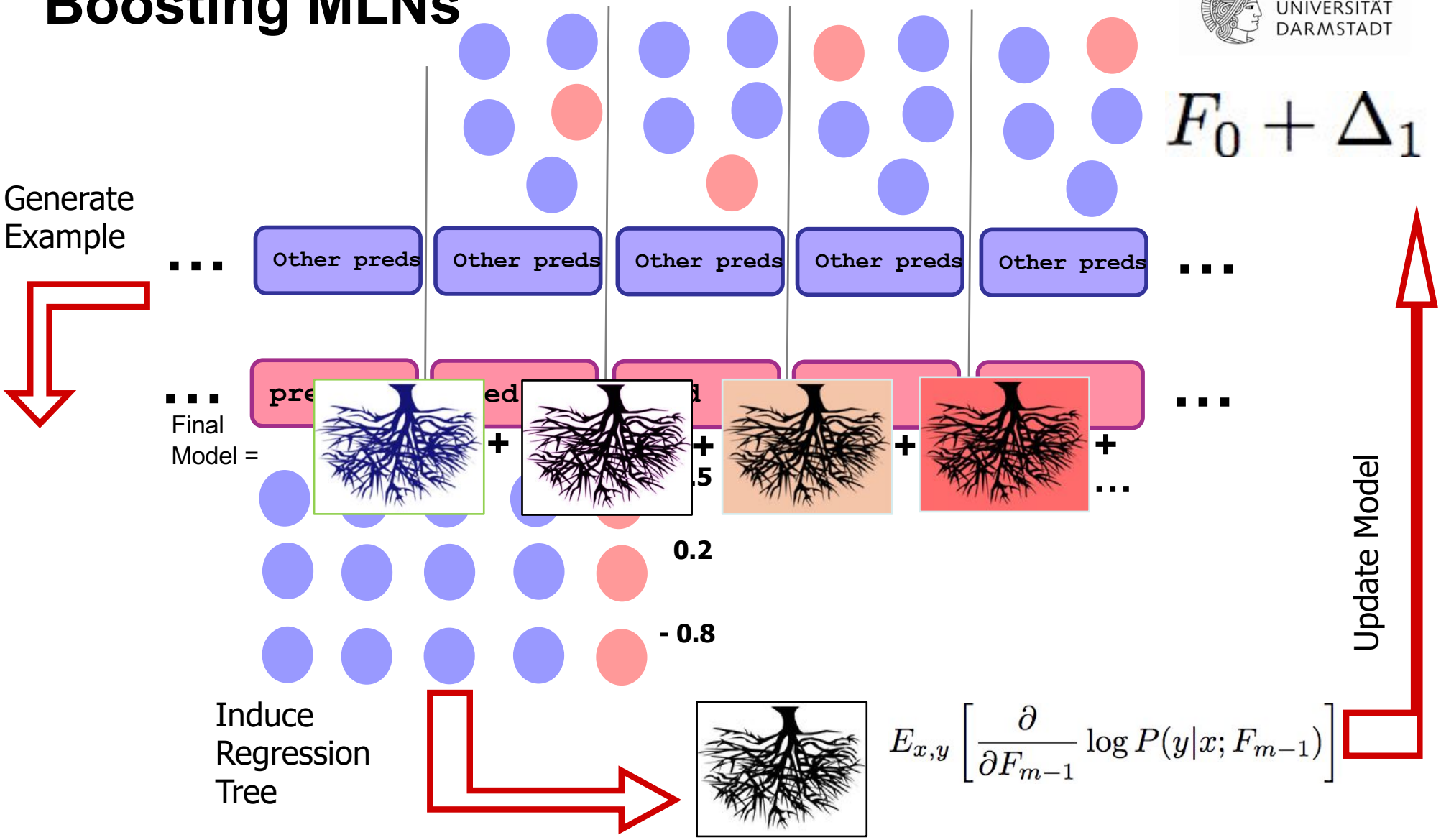
F_0

$$E_{x,y}\left[\frac{\partial}{\partial F_{m-1}} \log P(y|x; F_{m-1})\right]$$

Boosting MLNs



Boosting MLNs



$$F_0 + \Delta_1$$

$$E_{x,y}\left[\frac{\partial}{\partial F_{m-1}} \log P(y|x; F_{m-1})\right]$$

And recall that this applies to mathematical programs, too!

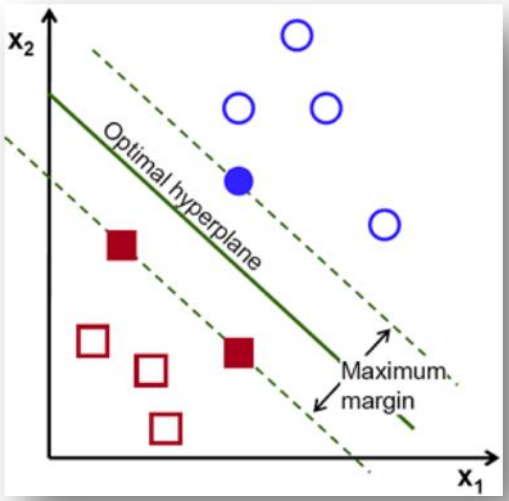
Write down SVM in „paper form.“ The machine compiles it into solver form.

```
#QUADRATIC OBJECTIVE
minimize: sum{J in feature(I,J)} weight(J)**2 + c1 * slack + c2 * coslack;

#labeled examples should be on the correct side
subject to forall {I in labeled(I)}: labeled(I)*predict(I) >= 1 - slack(I);

#slacks are positive
subject to forall {I in labeled(I)}: slack(I) >= 0;
```

Embedded within Python s.t. loops and rules can be used

reloop

RELOOP: A Toolkit for Relational Convex Optimization

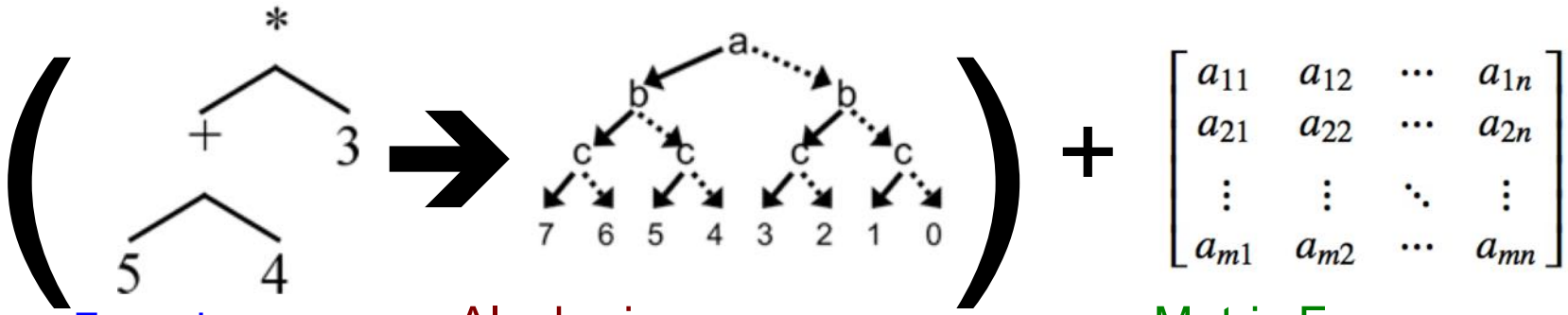


Support Vector Machines
Cortes, Vapnik MLJ 20(3):273-297, 1995

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

New field: Symbolic-numerical AI

TECHNISCHE UNIVERSITÄT DARMSTADT



$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

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

What have we learnt in Part II?

- Main insight for parameter estimation: parameter tighting
- Vanilla relational learning approach does a greedy search by adding/deleting literals/clauses using some (probabilistic) scoring function
- Learning many weak rules of how to change a model can be much faster
- Covers the whole AI spectrum

Conclusions: This “Deep AI ” excites industry:

RelationalAI, LogicBlox, Apple, and Uber are investing hundreds of millions of US dollars

And it appears in industrial strength solvers such as CPLEX and GUROBI

And there is a popular science books about it.

In 2016 Bill Gates recommended the book, alongside Nick Bostrom's *Superintelligence*, as one of two books everyone should read to understand AI.