UCLA Computer Science

STAR AI RESEARCH LAB UCLA

From Probabilistic Circuits to Probabilistic Programs and Back

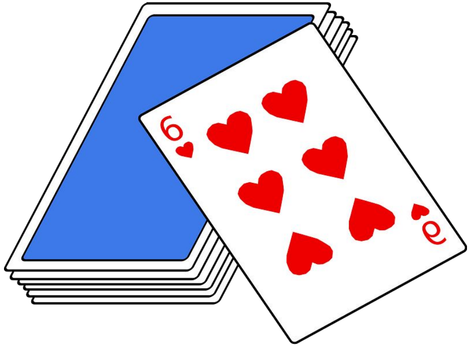
Guy Van den Broeck

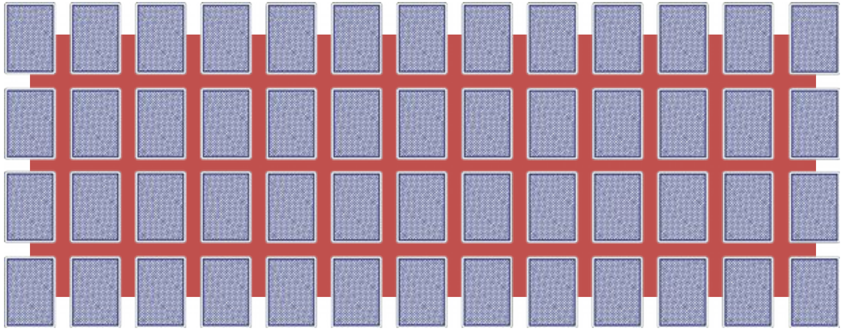
PROBPROG - Oct 24, 2020

Trying to be provocative

Probabilistic graphical models is how we do probabilistic AI!

Graphical models of variable-level (in)dependence
are a broken abstraction.





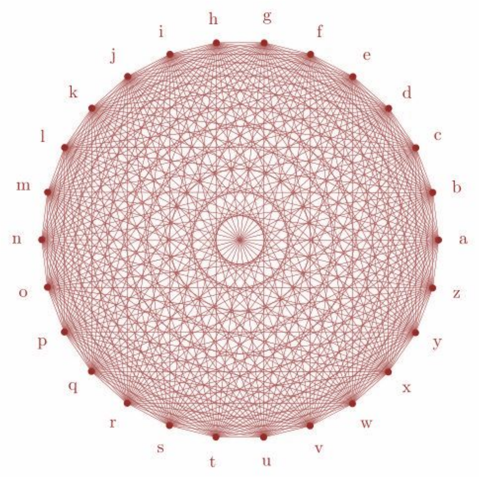
[VdB KRR15]

Trying to be provocative

Probabilistic graphical models is how we do probabilistic AI!

Graphical models of variable-level (in)dependence are a broken abstraction.

3.14 Smokes(x) ∧ Friends(x,y) ⇒ Smokes(y)



[VdB KRR15]

Trying to be provocative

Probabilistic graphical models is how we do probabilistic AI!

Graphical models of variable-level (in)dependence are a broken abstraction.

Bean Machine

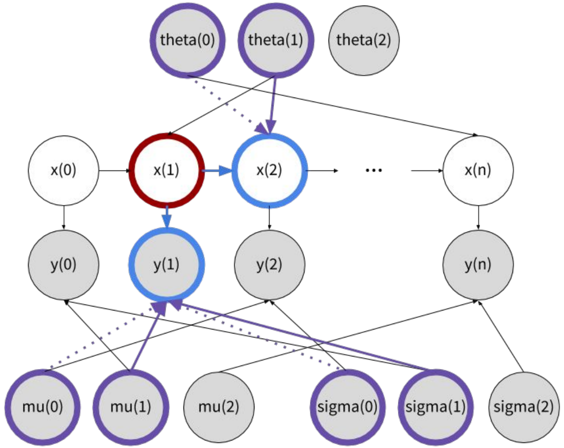
$$\mu_k \sim \text{Normal}(\alpha, \beta)$$

$$\sigma_k \sim \text{Gamma}(\nu, \rho)$$

$$\theta_k \sim \text{Dirichlet}(\kappa)$$

$$x_i \sim \begin{cases} \text{Categorical}(init) & \text{if } i = 0 \\ \text{Categorical}(\theta_{x_{i-1}}) & \text{if } i > 0 \end{cases}$$

$$y_i \sim \text{Normal}(\mu_{x_i}, \sigma_{x_i})$$



[Tehrani et al. PGM20]

Computational Abstractions

Let us think of probability distributions as objects that are computed.

Abstraction = Structure of Computation

Two examples:

2. Probabilistic Programs

Computational Abstractions

Let us think of probability distributions as objects that are computed.

Abstraction = Structure of Computation

Two examples:

1. Probabilistic Circuits
2. Probabilistic Programs

Probabilistic Circuits

Fully factorized

NaiveBayes *AndOrGraphs* *PDGs*

Trees *PSDDs* *CNets* *LTMs* *SPNs* *NADEs*

Thin Junction Trees *ACs* *MADEs* *MAFs* *VAEs*

DPPs *FVSBNs* *TACs* *IAFs* *NAFs* *RAEs*

Mixtures *BNs* *NICE* *FGs* *GANs*

RealNVP *MNs*

The Alphabet Soup of probabilistic models

Fully factorized

NaiveBayes AndOrGraphs PDGs

Trees PSDDs CNets LTMs SPNs NADEs

Thin Junction Trees ACs MADEs MAFs VAEs

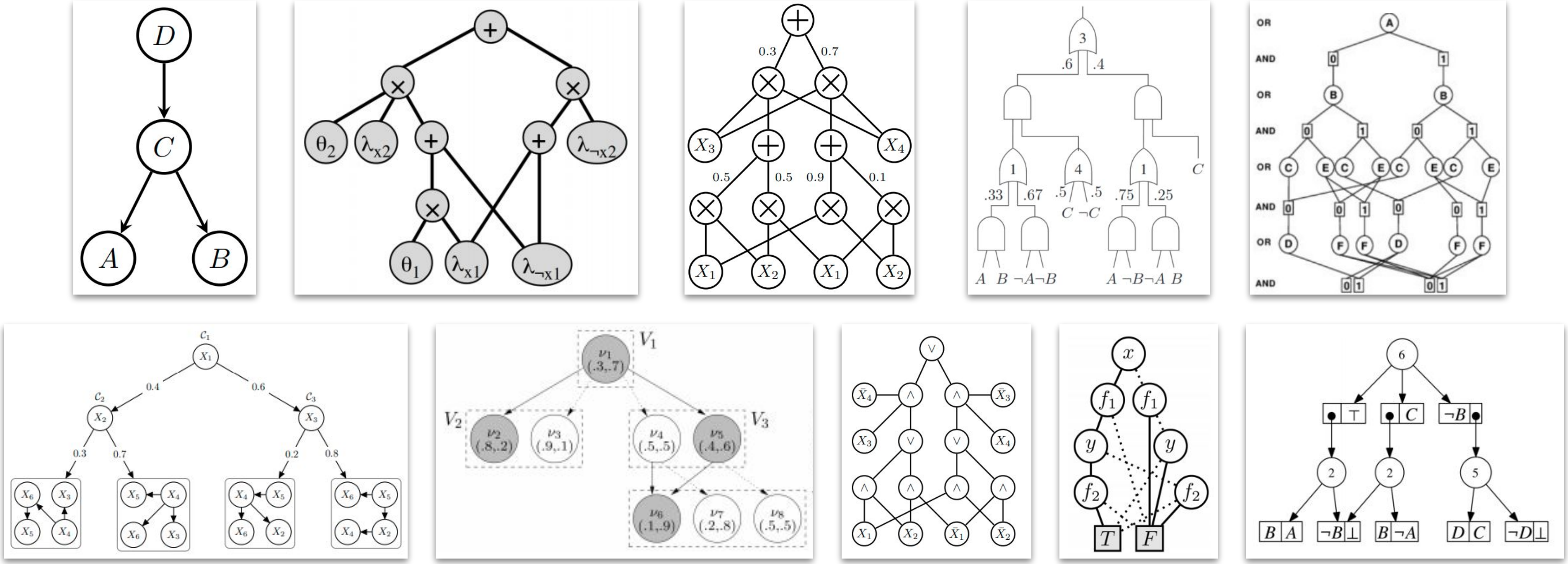
DPPs FVSBNs TACs IAFs NAFs RAEs

Mixtures BNs NICE FGs GANs

RealNVP MNs

Intractable and *tractable* models

Tractable Probabilistic Models



"*Every talk needs a joke and a literature overview slide, not necessarily distinct*"

- after Ron Graham

Fully factorized

NaiveBayes AndOrGraphs PDGs

Trees PSDDs CNets LTMs SPNs NADEs

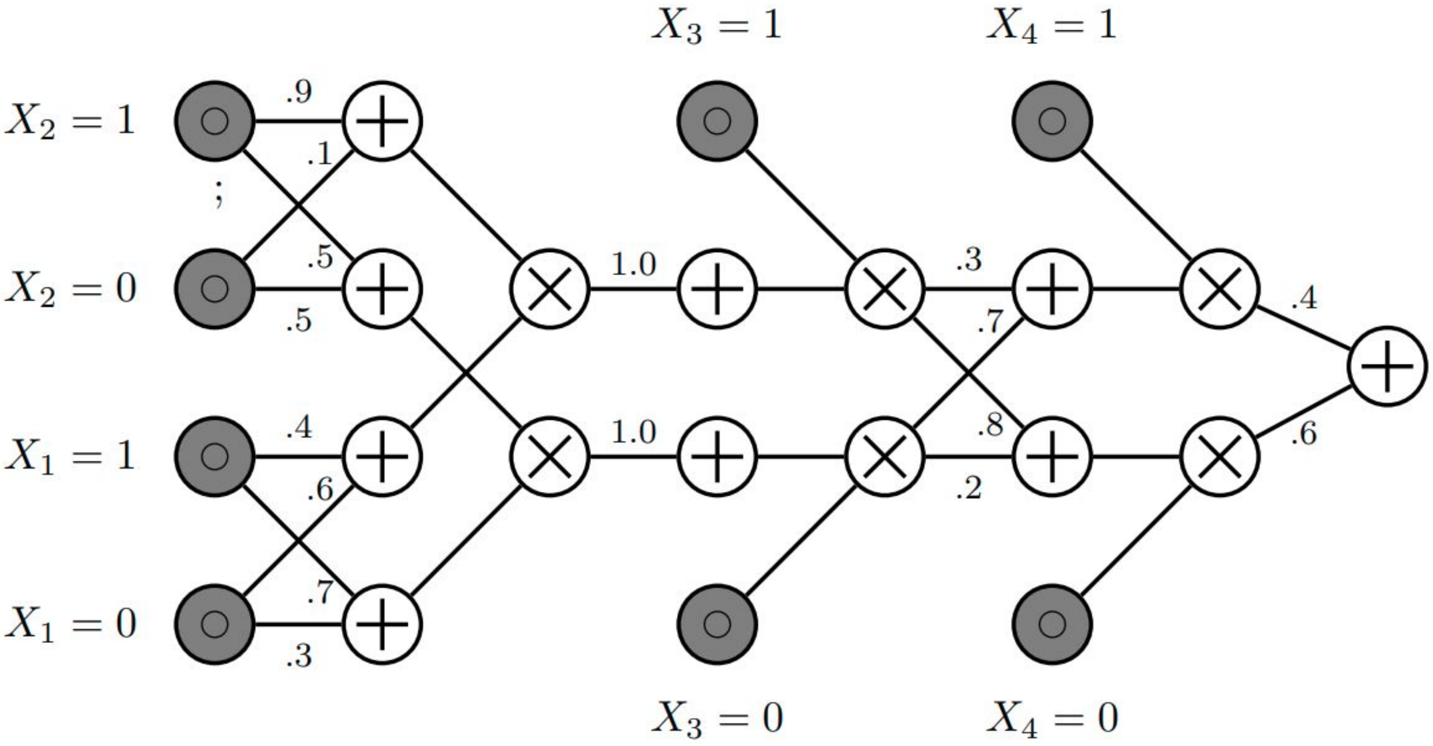
Thin Junction Trees ACs MADEs MAFs VAEs

DPPs FVSBNs TACs IAFs NAFs RAEs

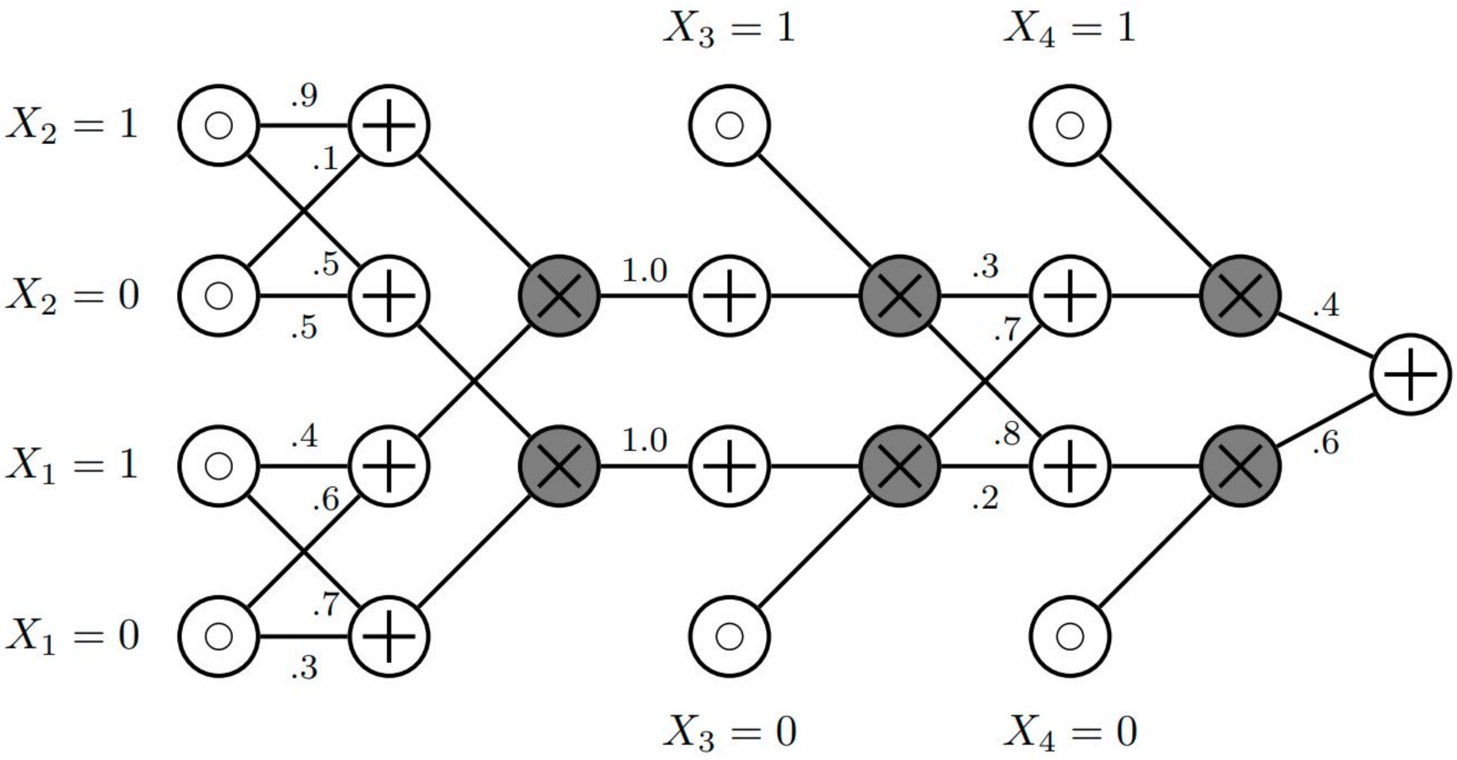
Mixtures BNs NICE FGs GANs

RealNVP MNs

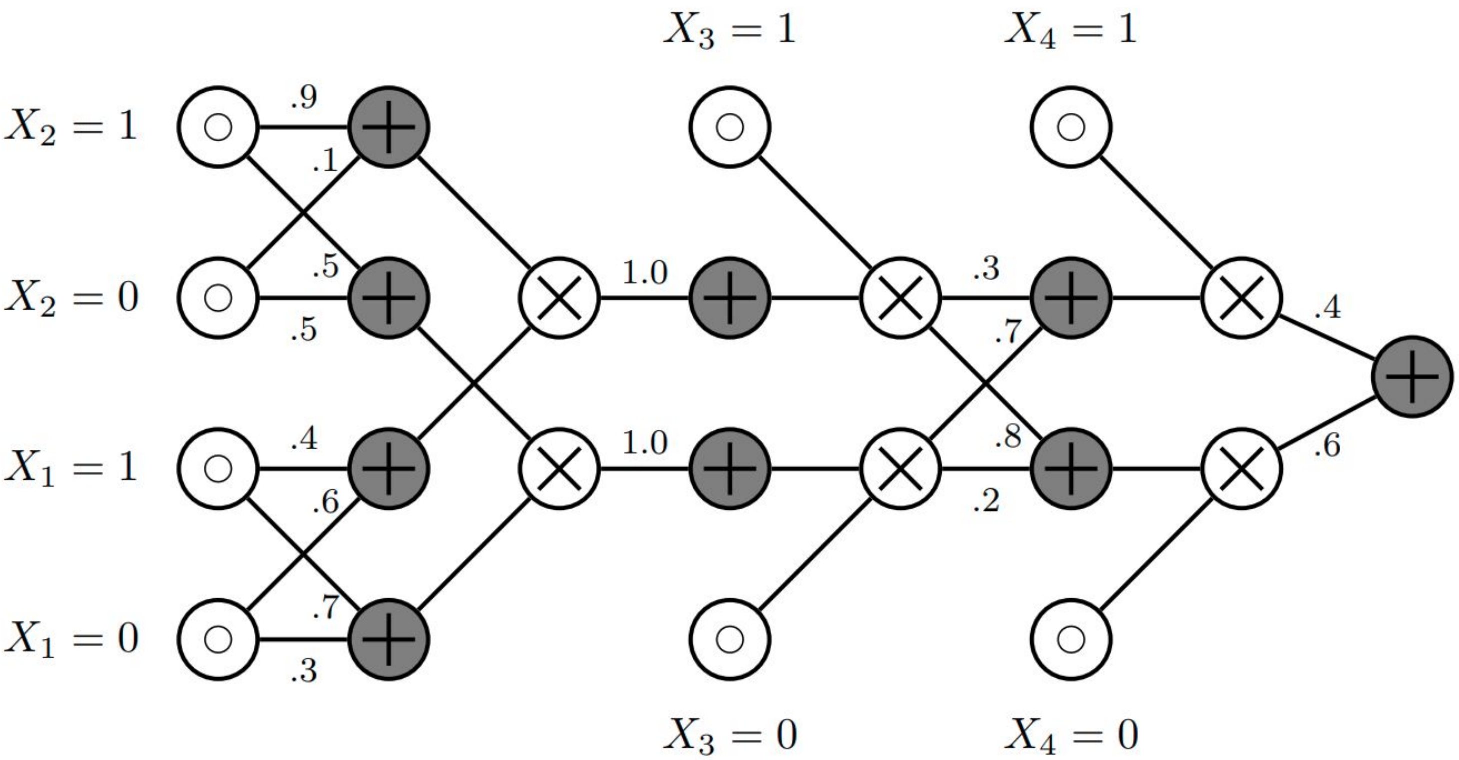
a *unifying framework* for tractable models



Input nodes c are tractable (simple) distributions,
e.g., univariate gaussian or indicator $p_c(X=1) = [X=1]$



Product nodes are factorizations $\prod_{c \in \mathsf{in}(n)} \mathrm{p}_c(\mathbf{x})$



Sum nodes are mixture models $\sum_{c \in \mathsf{in}(n)} \theta_{n,c}\, \mathrm{p}_c(\mathbf{x})$

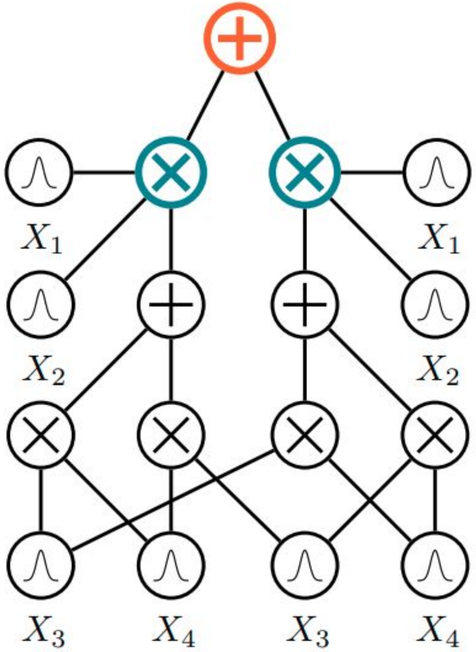
Smoothness + decomposability = tractable MAR

If $p(\mathbf{x}) = \sum_i w_i p_i(\mathbf{x})$, (***smoothness***):

$$\int p(\mathbf{x})d\mathbf{x} = \int \sum_i w_i p_i(\mathbf{x})d\mathbf{x} =$$

$$= \sum_i w_i \int p_i(\mathbf{x})d\mathbf{x}$$

$\Rightarrow$ *integrals are "pushed down" to children*



[Darwiche & Marquis JAIR 2001, Poon & Domingos UAI11]

Smoothness + **decomposability** = **tractable MAR**

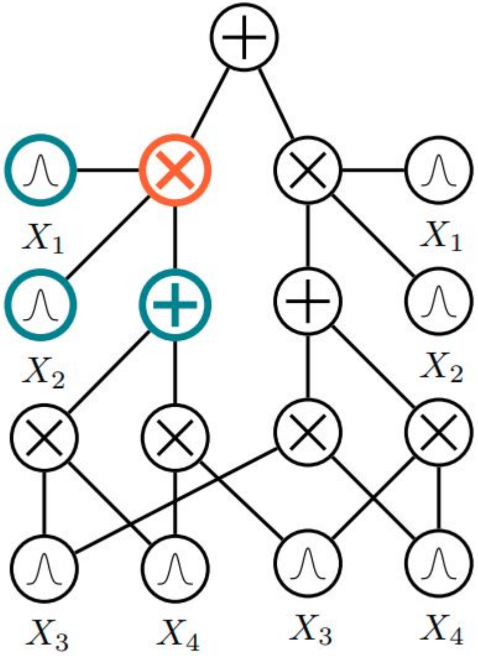
If $p(\mathbf{x}, \mathbf{y}, \mathbf{z}) = p(\mathbf{x})p(\mathbf{y})p(\mathbf{z})$, (***decomposability***):

$$\int\int\int p(\mathbf{x}, \mathbf{y}, \mathbf{z})d\mathbf{x}d\mathbf{y}d\mathbf{z} =$$

$$= \int\int\int p(\mathbf{x})p(\mathbf{y})p(\mathbf{z})d\mathbf{x}d\mathbf{y}d\mathbf{z} =$$

$$= \int p(\mathbf{x})d\mathbf{x} \int p(\mathbf{y})d\mathbf{y} \int p(\mathbf{z})d\mathbf{z}$$

$\Rightarrow$ *integrals decompose into easier ones*



Smoothness + decomposability = tractable MAR

Forward pass evaluation for MAR

$\Rightarrow$ *linear in circuit size!*

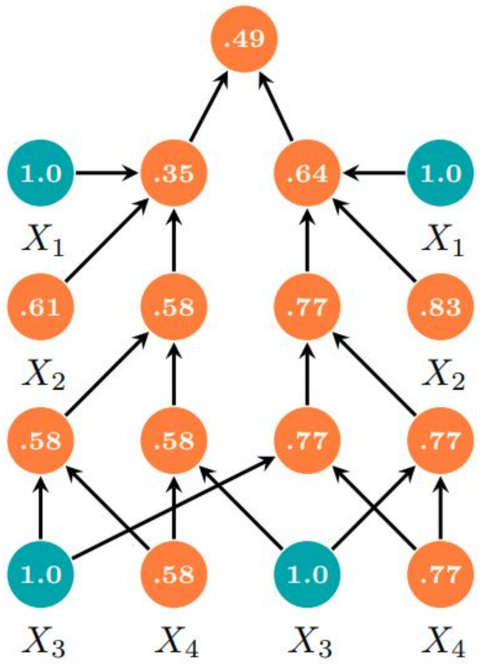
E.g. to compute $p(x_2, x_4)$:

- leafs over X_1 and X_3 output $\boldsymbol{Z}_i = \int p(x_i)dx_i$

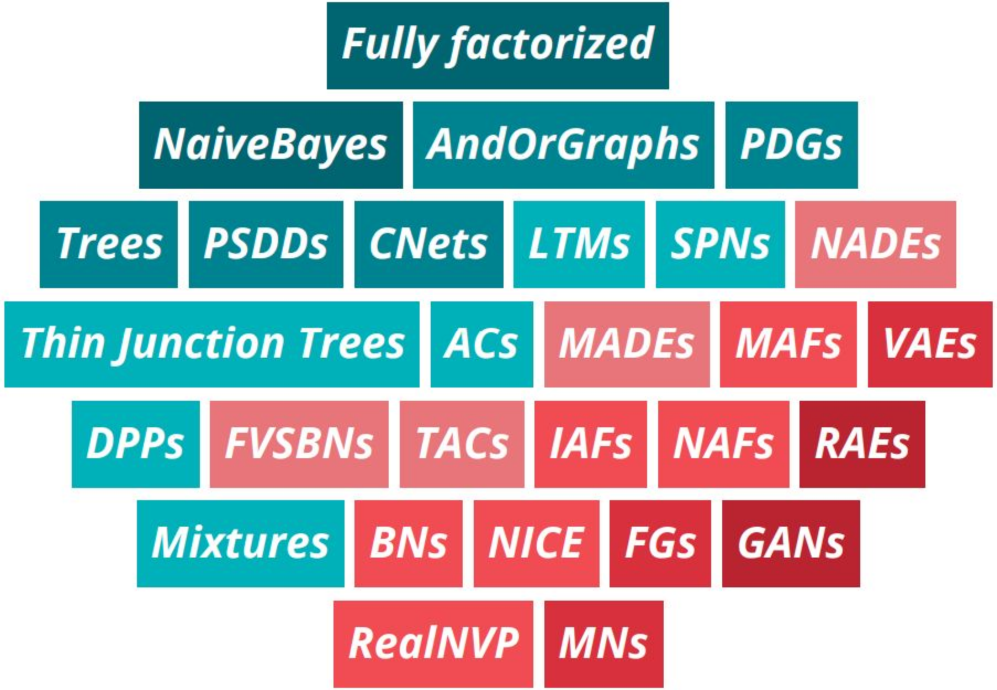
  $\Rightarrow$ *for normalized leaf distributions:* 1.0

- leafs over X_2 and X_4 output ***EVI***

- feedforward evaluation (bottom-up)



$\mathcal{P}$ \ $\mathcal{Q}$:	MAR marginal queries	CON conditional queries	MOM moments (mean,...)	MAP maximum a posteriori	MMAP marginal MAP	ENT entropy	DIV divergences (KLD,...)	EXP expected predictions
smoothness SMO	✓	✓	✓	✗	✗	✓	✓	✓
decomposability DEC	✓	✓	✓	✗	✗	✗	✗	✗
consistency CON	✗	✗	✗	✓	✓	✗	✗	✗
determinism DET	✗	✗	✗	✓	✗	✗	✗	✗
marginal determinism MAR-DET	✗	✗	✗	✗	✓	✓	✓	✗
structured decomposability STR-DEC	✗	✗	✗	✗	✗	✓	✗	✗
paired str. decomposability P-STR-DEC	✗	✗	✗	✗	✗	✗	✓	✓



tractability is a spectrum

	smooth	*dec.*	det.	str.dec.
Arithmetic Circuits (ACs) *[Darwiche 2003]*	✔	✔	✔	✘
Sum-Product Networks (SPNs) *[Poon et al. 2011]*	✔	✔	✘	✘
Cutset Networks (CNets) *[Rahman et al. 2014]*	✔	✔	✔	✘
Probabilistic Decision Graphs *[Jaeger 2004]*	✔	✔	✔	✔
(Affine) ADDs *[Hoey et al. 1999; Sanner et al. 2005]*	✔	✔	✔	✔
AndOrGraphs *[Dechter et al. 2007]*	✔	✔	✔	✔
PSDDs *[Kisa et al. 2014a]*	✔	✔	✔	✔

Fully factorized

NaiveBayes *AndOrGraphs* *PDGs*

Trees *PSDDs* *CNets* *LTMs* *SPNs* *NADEs*

Thin Junction Trees *ACs* *MADEs* *MAFs* *VAEs*

DPPs *FVSBNs* *TACs* *IAFs* *NAFs* *RAEs*

Mixtures *BNs* *NICE* *FGs* *GANs*

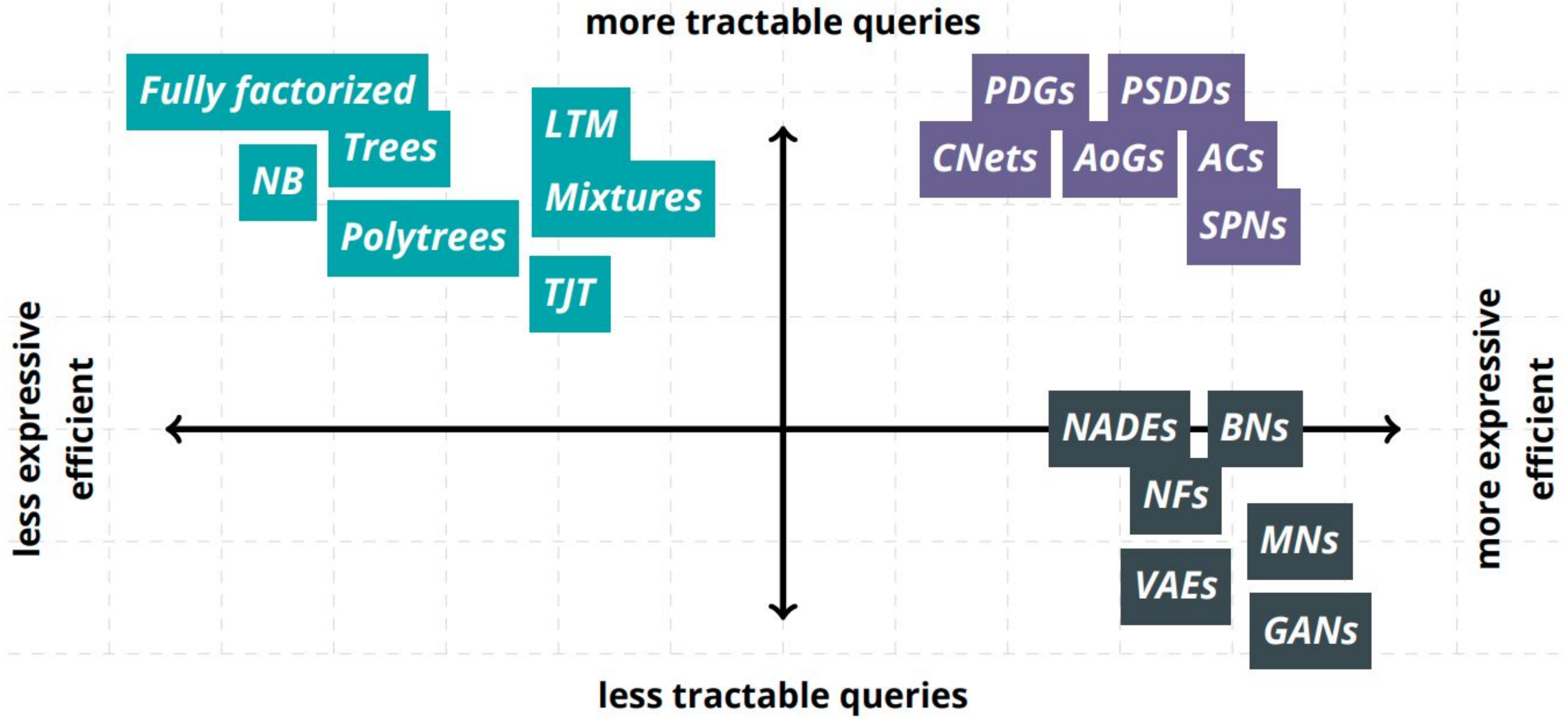
RealNVP *MNs*

Expressive models without *compromises*

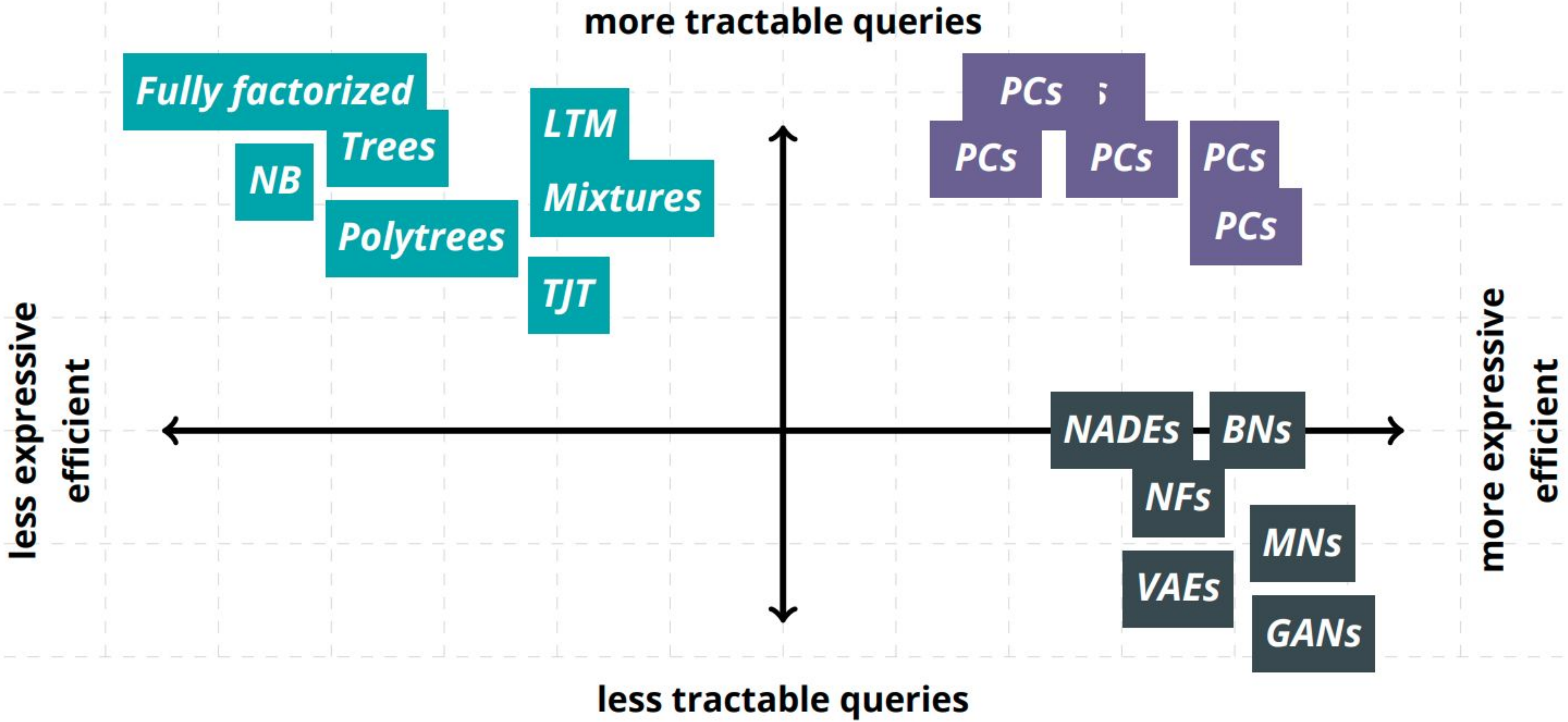
How expressive are probabilistic circuits?

density estimation benchmarks

dataset	best circuit	BN	MADE	VAE	dataset	best circuit	BN	MADE	VAE
nltcs	**-5.99**	-6.02	-6.04	**-5.99**	***dna***	**-79.88**	-80.65	-82.77	-94.56
msnbc	**-6.04**	**-6.04**	-6.06	-6.09	***kosarek***	**-10.52**	-10.83	-	-10.64
kdd	-2.12	-2.19	**-2.07**	-2.12	***msweb***	-9.62	-9.70	**-9.59**	-9.73
plants	**-11.84**	-12.65	-12.32	-12.34	***book***	-33.82	-36.41	-33.95	**-33.19**
audio	-39.39	-40.50	-38.95	**-38.67**	***movie***	-50.34	-54.37	-48.7	**-47.43**
jester	-51.29	**-51.07**	-52.23	-51.54	***webkb***	-149.20	-157.43	-149.59	**-146.9**
netflix	-55.71	-57.02	-55.16	**-54.73**	***cr52***	-81.87	-87.56	-82.80	**-81.33**
accidents	-26.89	**-26.32**	-26.42	-29.11	***c20ng***	-151.02	-158.95	-153.18	**-146.9**
retail	**-10.72**	-10.87	-10.81	-10.83	***bbc***	**-229.21**	-257.86	-242.40	-240.94
pumbs*	-22.15	**-21.72**	-22.3	-25.16	***ad***	-14.00	-18.35	**-13.65**	-18.81



more tractable queries
Fully factorized
Trees
NB
LTM
Mixtures
Polytrees
TJT
PDGs
PSDDs
CNets
AoGs
ACs
SPNs
less expressive
efficient
more expressive
efficient
NADEs
BNs
NFs
MNs
VAEs
GANs
less tractable queries

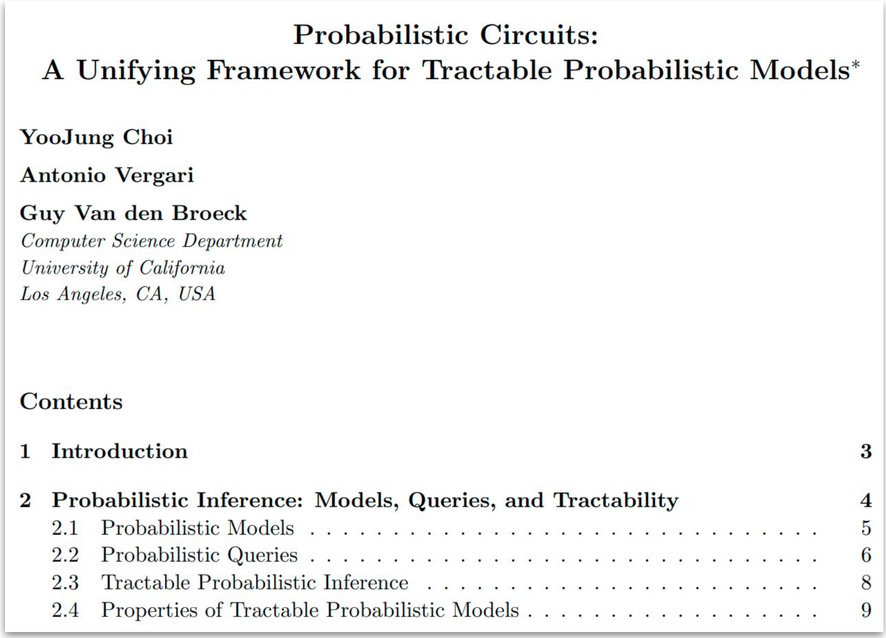

more tractable queries
Fully factorized
Trees
NB
Polytrees
LTM
Mixtures
TJT
PCs
PCs
PCs
PCs
PCs
less expressive
efficient
more expressive
efficient
NADEs
BNs
NFs
MNs
VAEs
GANs
less tractable queries

Want to learn more?

Tutorial (3h)

https://youtu.be/2RAG5-L9R70

Overview Paper (80p)



Probabilistic Circuits:
A Unifying Framework for Tractable Probabilistic Models*

YooJung Choi
Antonio Vergari
Guy Van den Broeck
Computer Science Department
University of California
Los Angeles, CA, USA

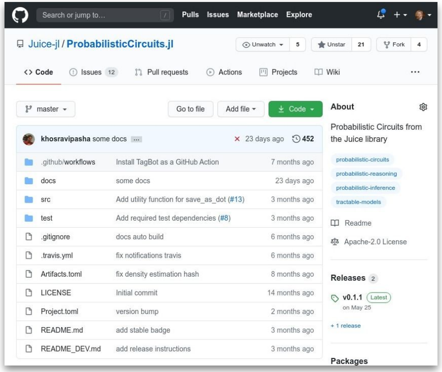
Contents

http://starai.cs.ucla.edu/papers/ProbCirc20.pdf

Training PCs in Julia with Juice.jl

Training maximum likelihood parameters of probabilistic circuits

```
julia> using ProbabilisticCircuits;
julia> data, structure = load(...);
julia> num_examples(data)
17,412
julia> num_edges(structure)
270,448
julia> @btime estimate_parameters(structure , data);
63 ms
```



Custom SIMD and CUDA kernels to parallelize over layers and training examples.

[https://github.com/Juice-jl/]

Probabilistic circuits seem awfully general.

Are all tractable probabilistic models probabilistic circuits?

Determinantal Point Processes (DPPs)

DPPs are models where probabilities are specified by (sub)determinants

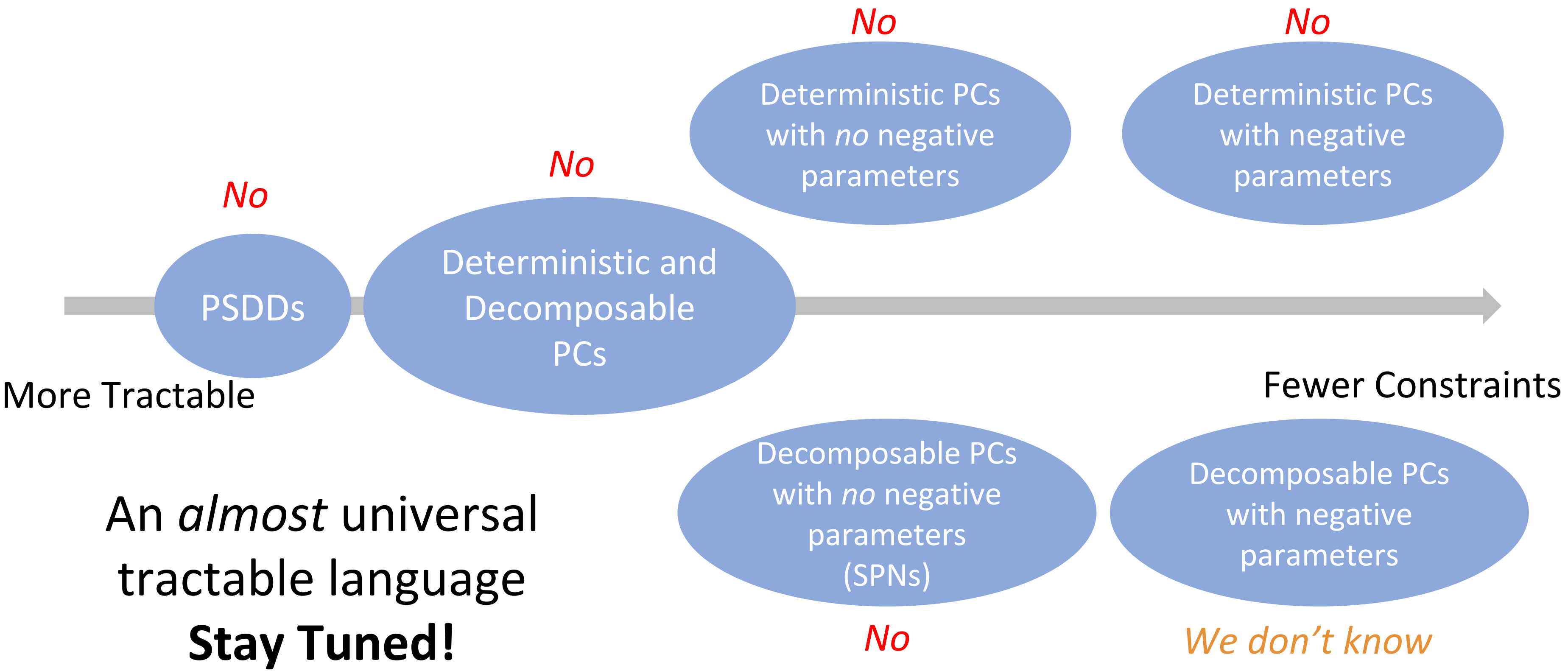
$$L = \begin{bmatrix} 1 & 0.9 & 0.8 & 0 \\ 0.9 & 0.97 & 0.96 & 0 \\ 0.8 & 0.96 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\Pr_L(X_1 = 1, X_2 = 0, X_3 = 1, X_4 = 0) = \frac{1}{\det(L + I)} \det(L_{\{1,2\}})$$

Computing marginal probabilities is *tractable.*

[Zhang et al. UAI20]

We cannot tractably represent DPPs with classes of PCs … yet



An *almost* universal tractable language

Stay Tuned!

[Zhang et al. UAI20; Martens & Medabalimi Arxiv15]

The AI Dilemma

Pure Logic **Pure Learning**

The AI Dilemma

Pure Logic

Pure Learning

- Slow thinking: deliberative, cognitive, model-based, extrapolation
- Amazing achievements until this day
- "*Pure logic is brittle*"

  noise, uncertainty, incomplete knowledge, …

The AI Dilemma

Pure Logic **Pure Learning**

- Fast thinking: instinctive, perceptive, model-free, interpolation
- Amazing achievements recently
- “*Pure learning is brittle*”

  bias, algorithmic fairness, interpretability, explainability, adversarial attacks, unknown unknowns, calibration, verification, missing features, missing labels, data efficiency, shift in distribution, general robustness and safety

  fails to incorporate a sensible model of the world

Pure Logic **Probabilistic World Models** **Pure Learning**

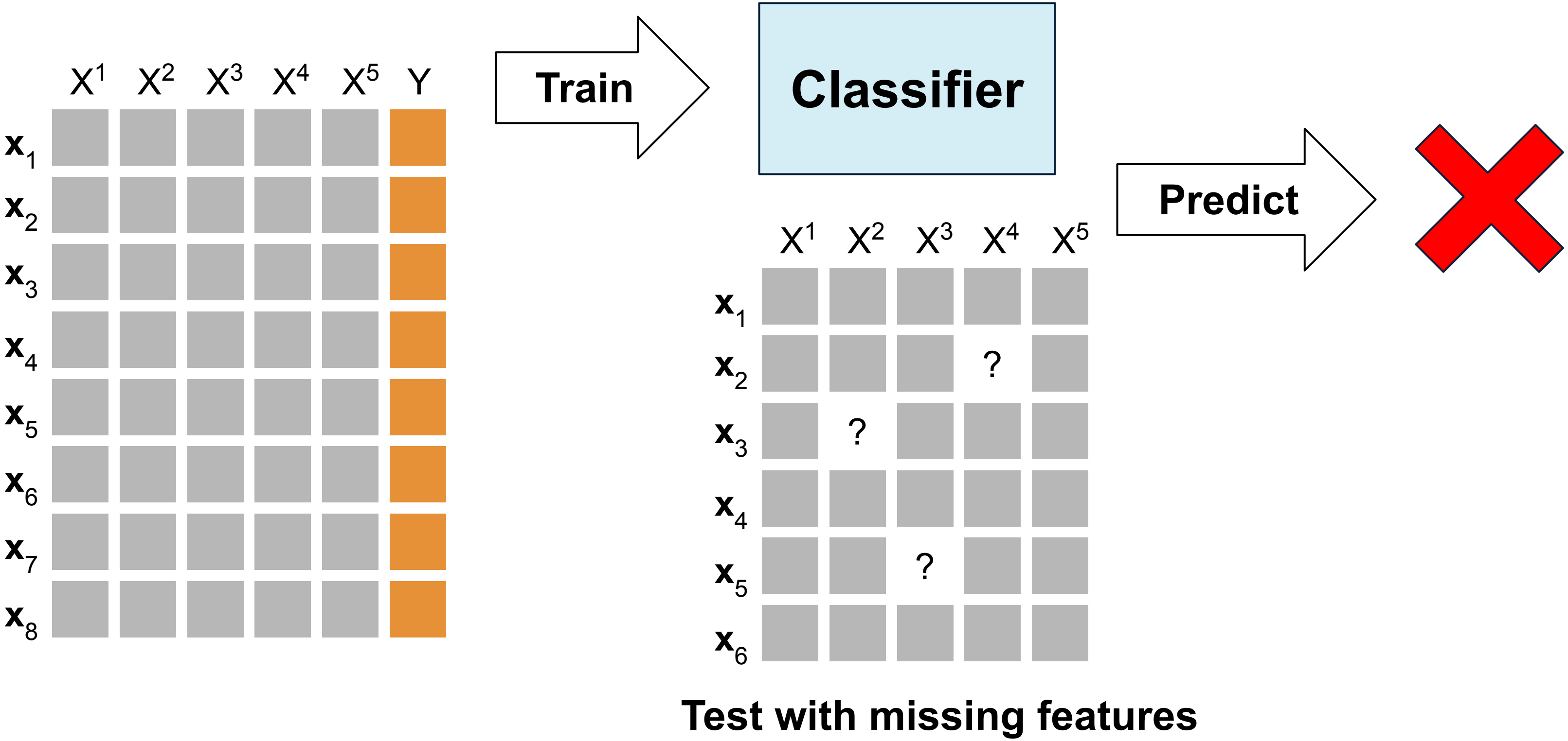
A New Synthesis of Learning and Reasoning

- "*Pure learning is brittle*"

  bias, **algorithmic fairness**, interpretability, **explainability**, adversarial attacks, unknown unknowns, calibration, verification, **missing features**, missing labels, data efficiency, shift in distribution, general robustness and safety

  fails to incorporate a sensible model of the world

Prediction with Missing Features



Expected Predictions

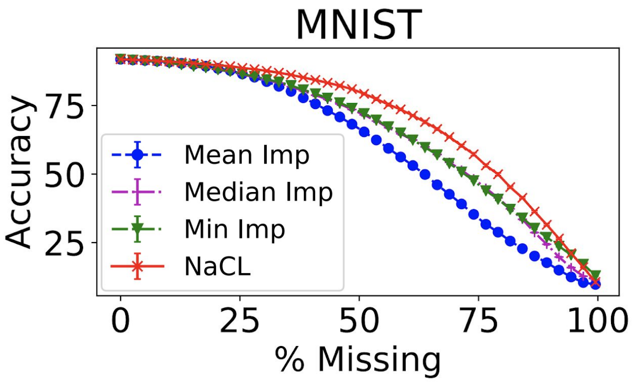
Consider **all possible complete inputs** and **reason** about the *expected* behavior of the classifier

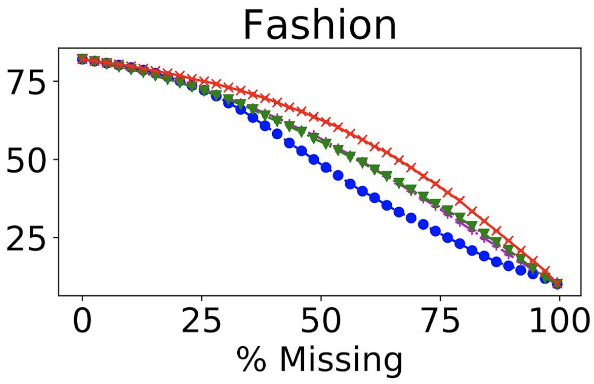
$$\mathbb{E}_{\mathbf{x}^m \sim p(\mathbf{x}^m|\mathbf{x}^o)}\left[f(\mathbf{x}^m\mathbf{x}^o)\right]$$

$\boldsymbol{x}^{\mathrm{o}}$ = observed features
$\boldsymbol{x}^{\mathrm{m}}$ = missing features

Experiment:

- *f(x)* = logistic regres.
- *p(x)* = naive Bayes





[Khosravi et al. IJCAI19, NeurIPS20, Artemiss20]

What about complex feature distributions?

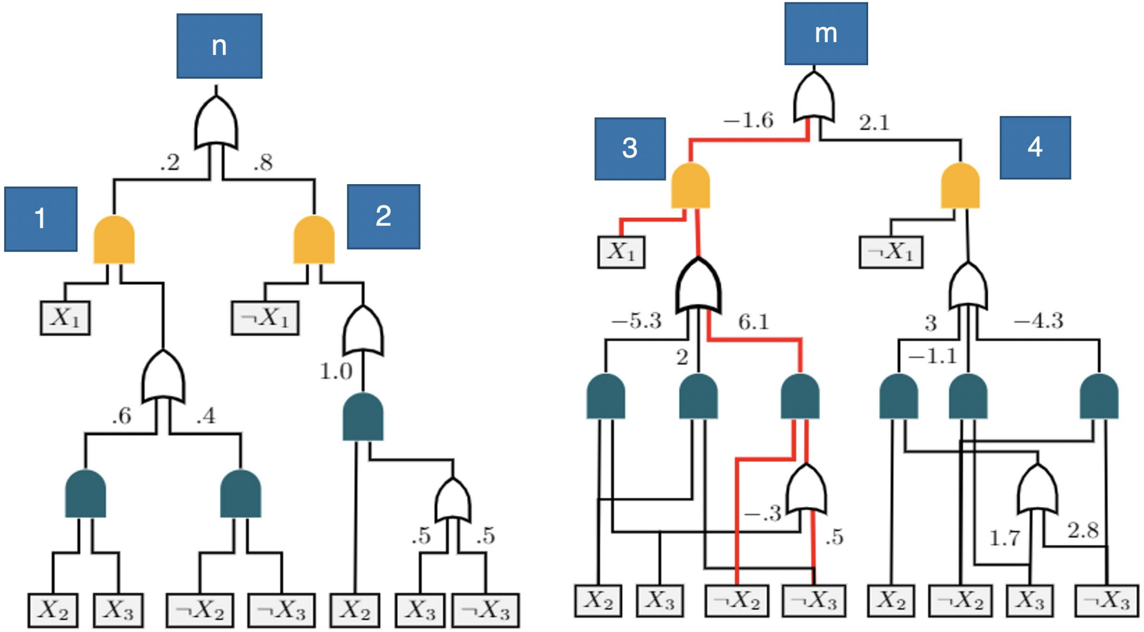
- feature distribution is a probabilistic circuits
- classifier is a compatible regression circuit

☺

Recursion that "breaks down" the computation.

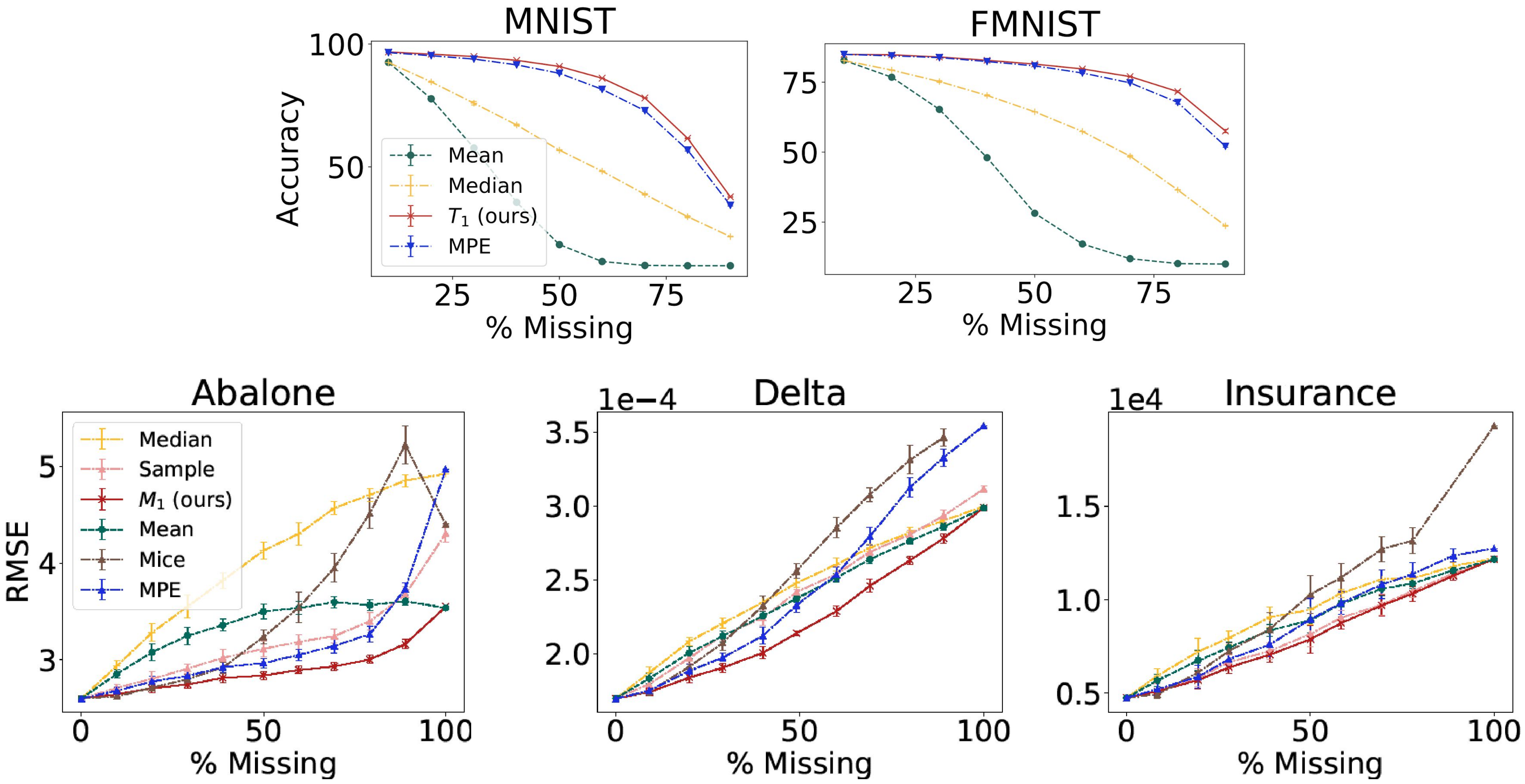
Expectation of function m w.r.t. dist. n ?

Solve subproblems: (1,3), (1,4), (2,3), (2,4)



[Khosravi et al. IJCAI19, NeurIPS20, Artemiss20]

Probabilistic Circuits for Missing Data



[Khosravi et al. IJCAI19, NeurIPS20, Artemiss20]

ADV inference in Julia with Juice.jl

```
using ProbabilisticCircuits
pc = load_prob_circuit(zoo_psdd_file("insurance.psdd"));
rc = load_logistic_circuit(zoo_lc_file("insurance.circuit"), 1);
```

Is the predictive model biased by gender?

```
groups = make_observations([["male"], ["female"]])
exps, _ = Expectation(pc, rc, groups);
println("Female  : \$ $(exps[2])");
println("Male    : \$ $(exps[1])");
println("Diff    : \$ $(exps[2] - exps[1])");
Female  : $ 14170.125469335406
Male    : $ 13196.548926381849
Diff    : $ 973.5765429535568
```

Model-Based Algorithmic Fairness: FairPC

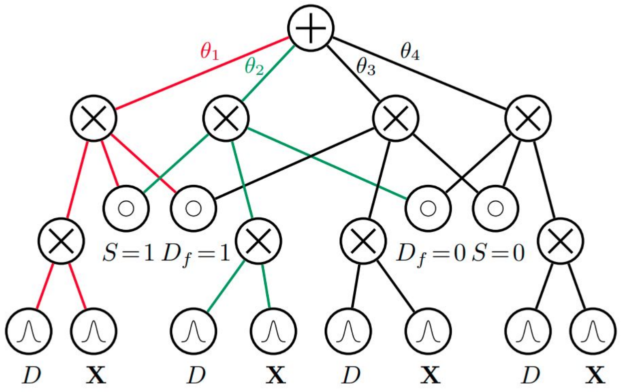
Learn classifier given

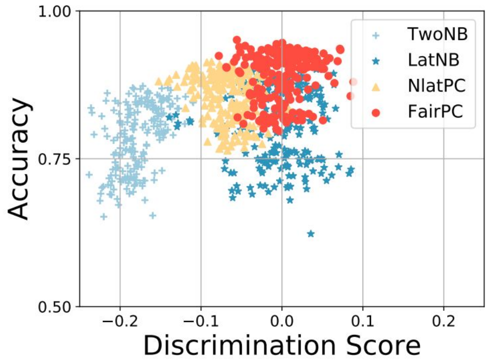
- features S and X
- training labels/decisions D

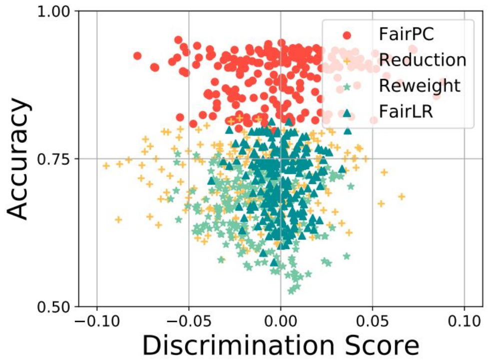
Group fairness by demographic parity:

Fair decision D_f should be independent of the sensitive attribute S

Discover the latent fair decision D_f by learning a PC.







[Choi et al. Arxiv20]

Probabilistic Sufficient Explanations

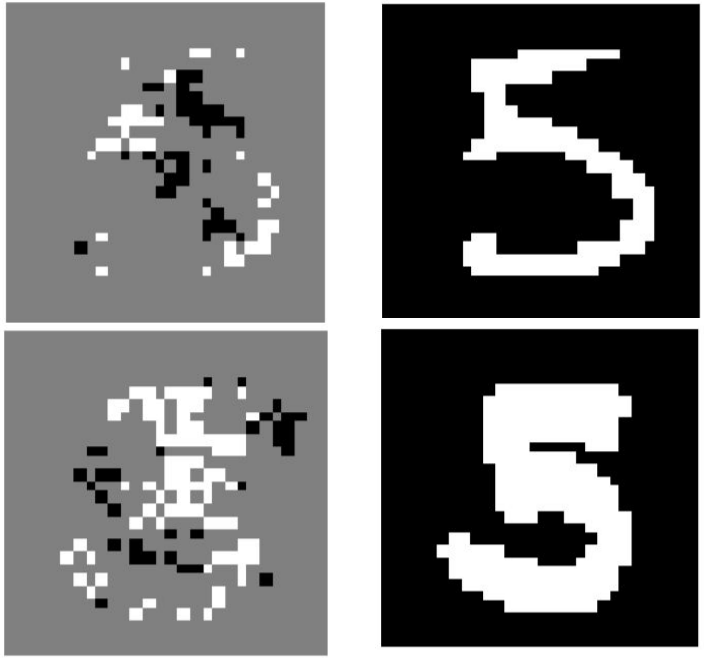
Goal: explain an instance of classification (a specific prediction)

Explanation is a subset of features, s.t.

1. The explanation is "probabilistically sufficient"

   Under the feature distribution, given the explanation, the classifier is likely to make the observed prediction.

2. It is minimal and "simple"



[Khosravi et al. IJCAI19, Wang et al. XXAI20]

Pure Logic **Probabilistic World Models** **Pure Learning**

A New Synthesis of Learning and Reasoning

"*Pure learning is brittle*"

bias, **algorithmic fairness**, interpretability, **explainability**, adversarial attacks, unknown unknowns, calibration, verification, **missing features**, missing labels, data efficiency, shift in distribution, general robustness and safety

We need to incorporate a sensible probabilistic model of the world

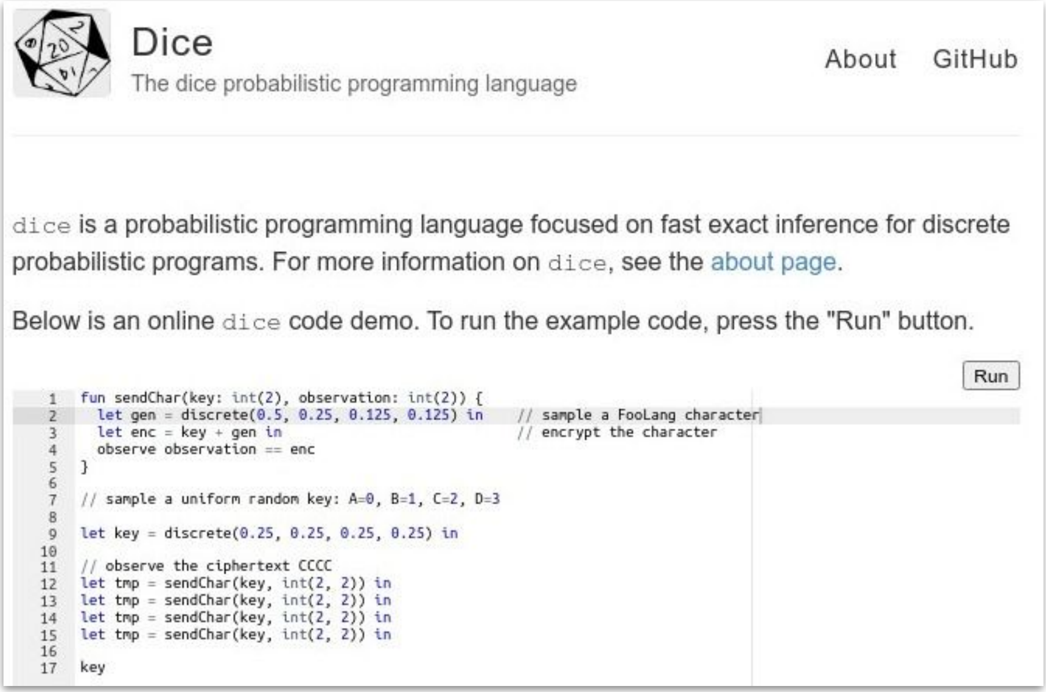
Probabilistic Programs

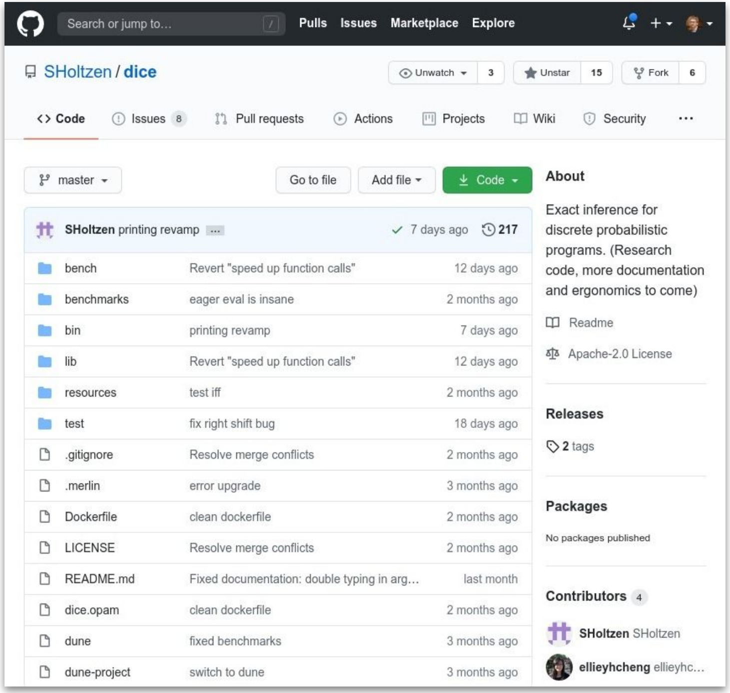
Dice probabilistic programming language

Talk in 25min

http://dicelang.cs.ucla.edu/

https://github.com/SHoltzen/dice

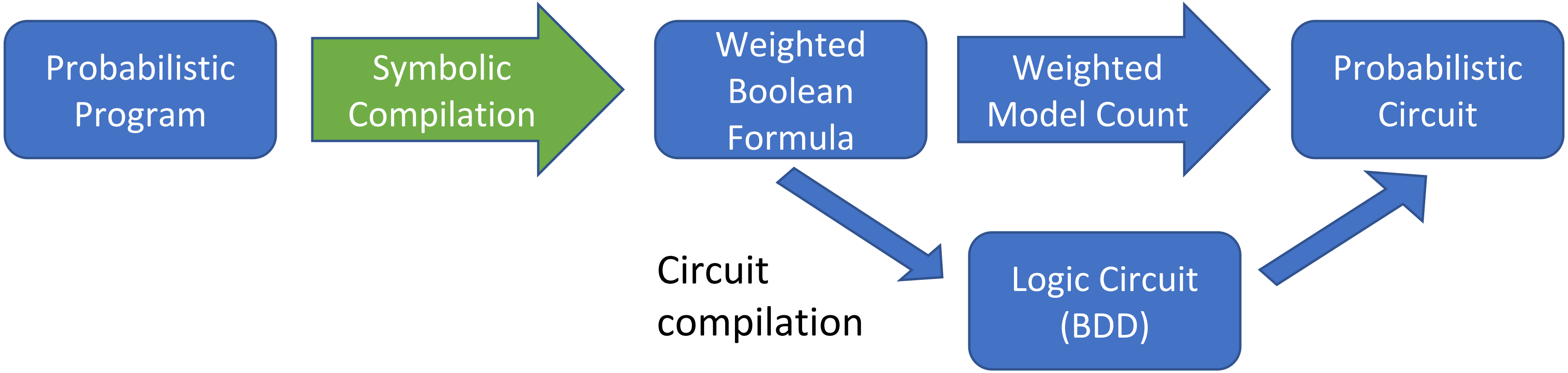




[Holtzen et al. OOPSLA20]

Symbolic Compilation to Probabilistic Circuits

Talk in 25min



State of the art for discrete probabilistic program inference!

Conclusions

- Are we already in the age of computational abstractions?
- **Probabilistic circuits** for learning deep tractable probabilistic models
- **Probabilistic programs** as the new probabilistic knowledge representation language
- Two computational abstractions go hand in hand

Thanks

My students/postdoc who did the real work are graduating.

There are some awesome people on the academic job market!