What Can We Compute from a Representation?
Structure and Composition for Tractable Reasoning

YooJung Choi

ICML 2026 Workshop on Connecting Low-rank Representations in AI – *July 11, 2026*

Data & Model Compression

Tensor Factorizations

Low-rank Adapters

*We exploit **structure** like low-rankness all the time*

Interpretability

Probabilistic Circuits

Factor Graphs, Graphical Models

Structure for tractable computation

We can exploit structure (low-rank, sparsity, etc) in data & models for …

Compact representations,

Efficient parameterization & training,

Efficient computation of models, …

but also for…

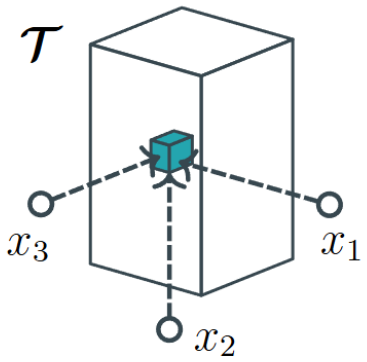
*Tractable **reasoning** about models*

Outline

1. Probabilistic circuits – How structure enables tractable inference
2. A compositional view of inference – How tractable inference algorithms can be derived systematically
3. Tractable reasoning for distributional robustness

Imagine a *probability distribution*

$$P(x_1, x_2, x_3)$$

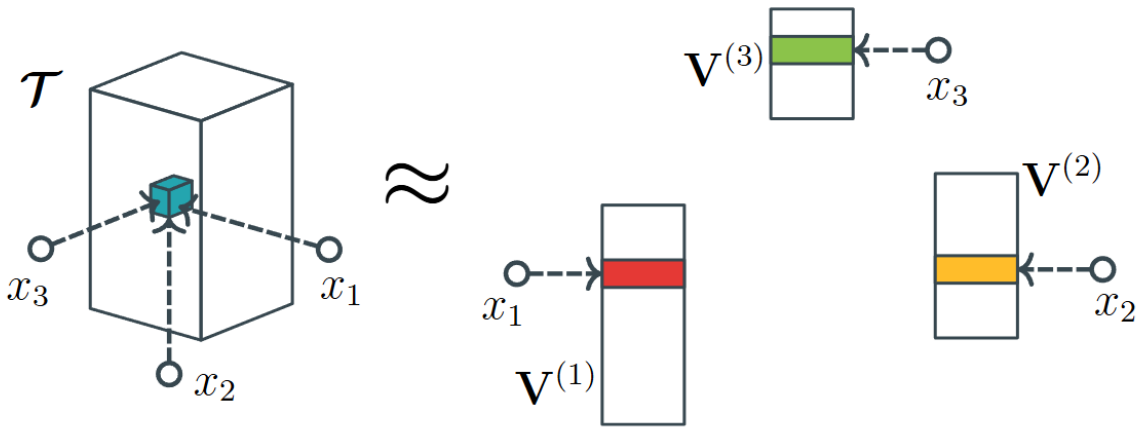


Tensor factorization

(e.g. CP)

Imagine a *probability distribution*

$P(x_1, x_2, x_3)$



$$t_{x_1 x_2 x_3} \approx \sum_{r=1}^{R} v^{(1)}_{x_1 r} \; v^{(2)}_{x_2 r} \; v^{(3)}_{x_3 r}$$

$$\mathbf{V}^{(1)} \in \mathbb{R}^{I_1 \times R}, \mathbf{V}^{(2)} \in \mathbb{R}^{I_2 \times R}, \mathbf{V}^{(3)} \in \mathbb{R}^{I_3 \times R}$$

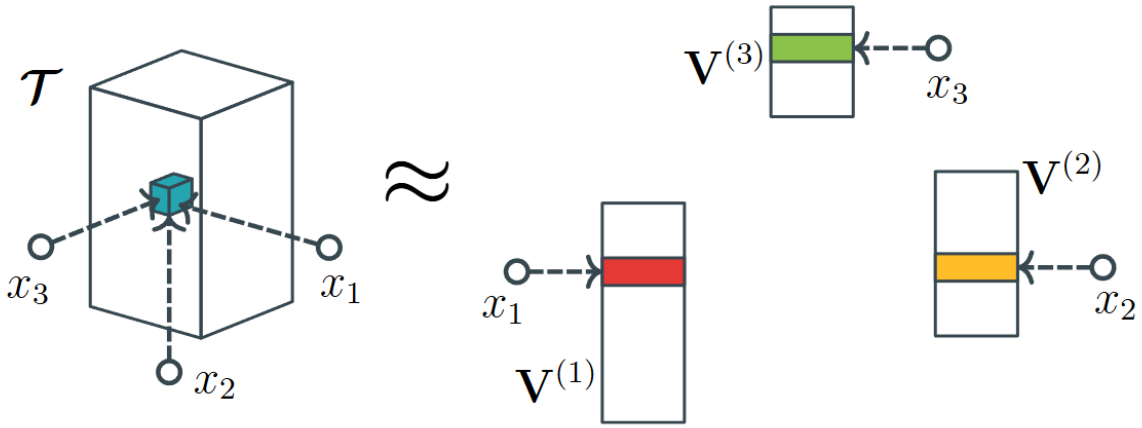
Imagine a *probability distribution*

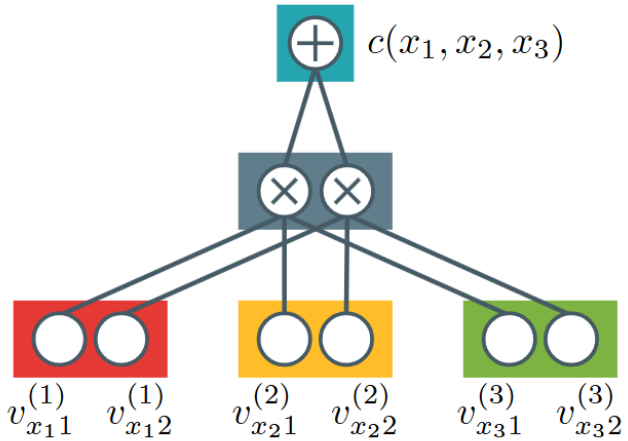
$P(x_1, x_2, x_3)$

Tensor factorization

(e.g. CP)

as a ***circuit***





$$t_{x_1 x_2 x_3} \approx \sum_{r=1}^{R} v^{(1)}_{x_1 r} \, v^{(2)}_{x_2 r} \, v^{(3)}_{x_3 r}$$

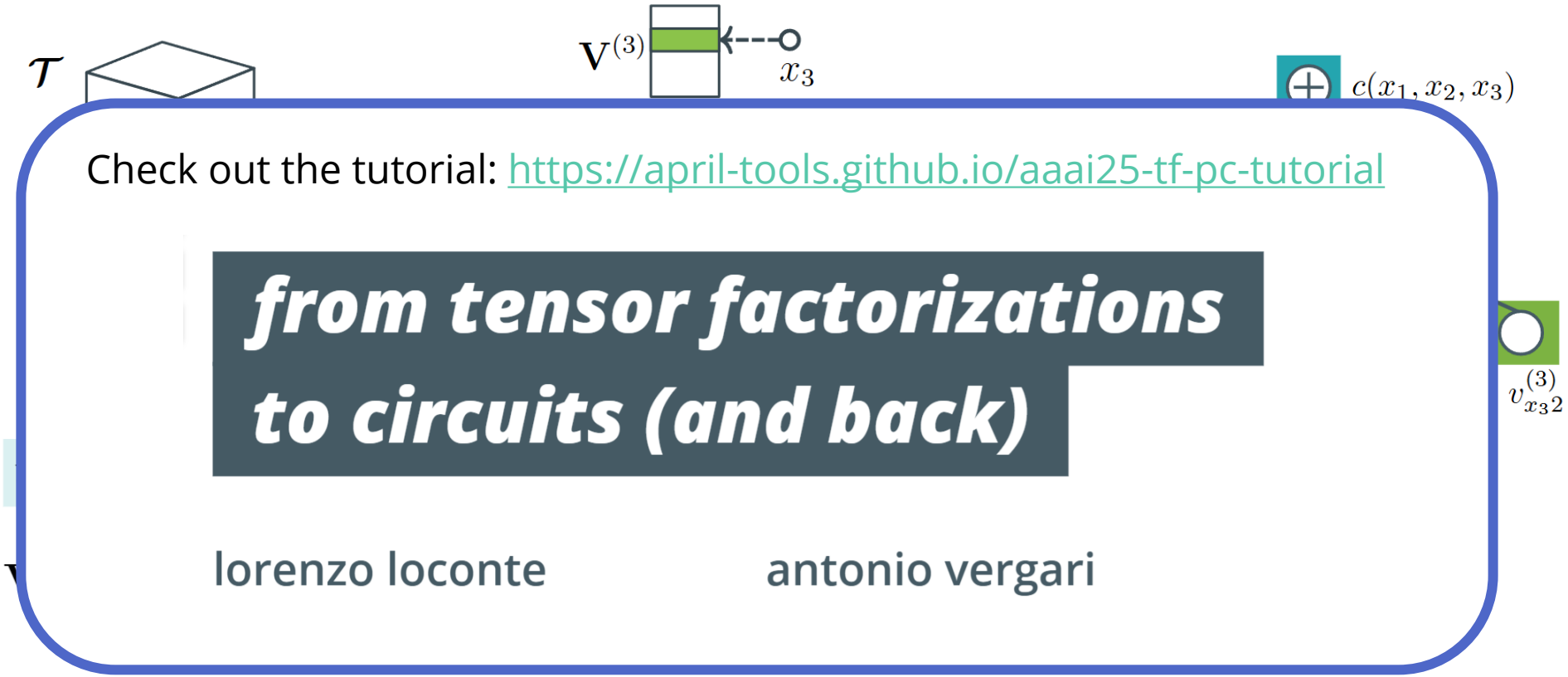
$$\mathbf{V}^{(1)} \in \mathbb{R}^{I_1 \times R}, \mathbf{V}^{(2)} \in \mathbb{R}^{I_2 \times R}, \mathbf{V}^{(3)} \in \mathbb{R}^{I_3 \times R}$$

Imagine a *probability distribution* $P(x_1, x_2, x_3)$

Tensor factorization

(e.g. CP)

as a ***circuit***



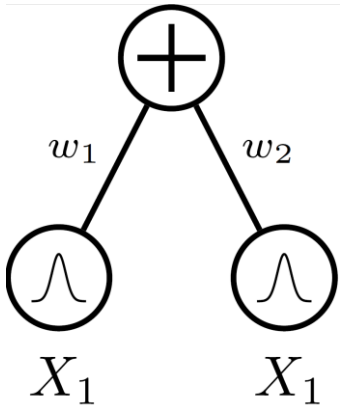
Check out the tutorial: https://april-tools.github.io/aaai25-tf-pc-tutorial

from tensor factorizations to circuits (and back)

lorenzo loconte antonio vergari

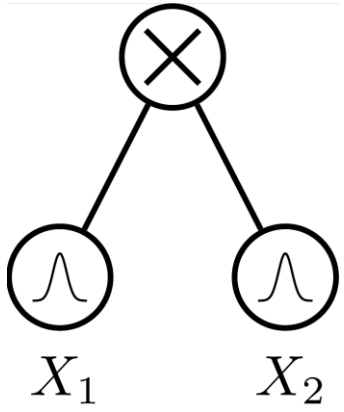
Probabilistic circuits (PCs)

computational graphs that recursively define probability mass/density functions



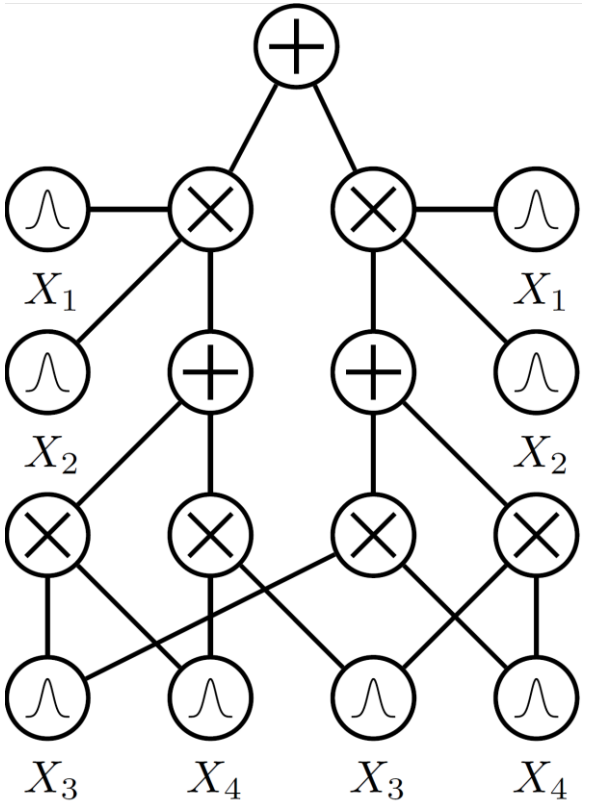
$$p(X_1) = w_1 p_1(X_1) + w_2 p_2(X_1)$$

⇒ mixtures



$$p(X_1, X_2) = p(X_1) \cdot p(X_2)$$

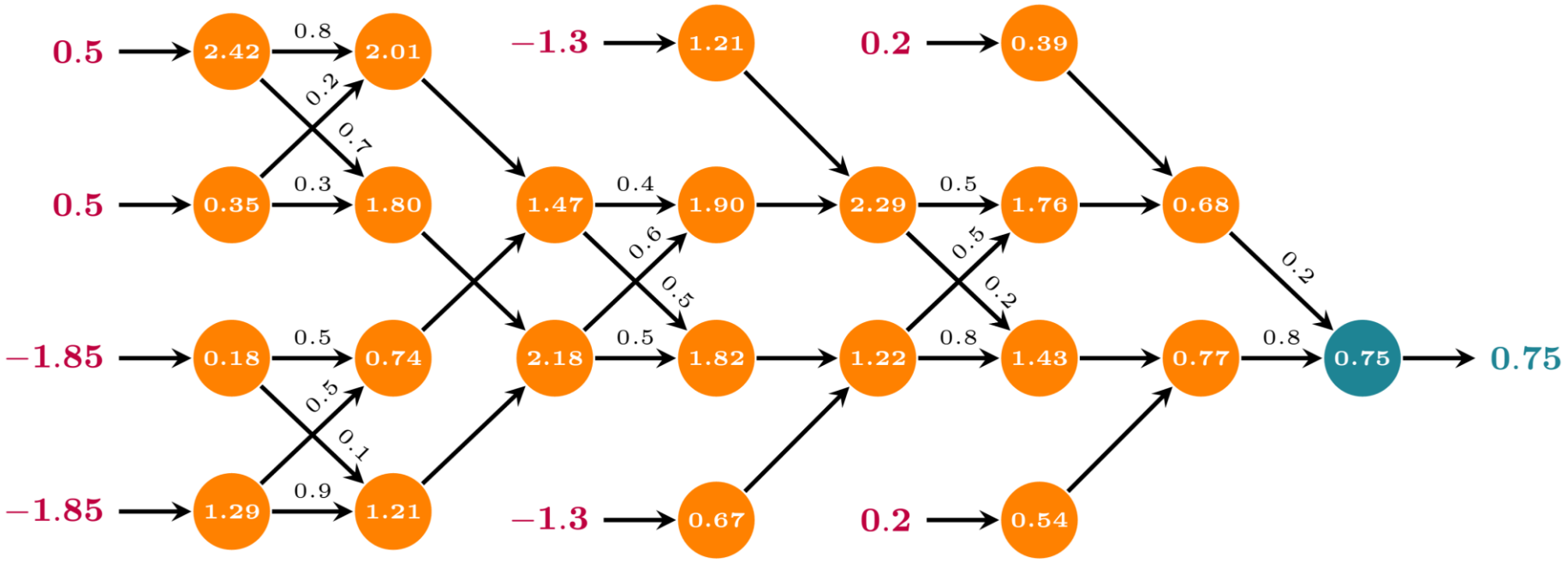
⇒ factorizations



Probabilistic circuits (PCs)

Compute $p(\mathbf{X})$ by simple feedforward evaluation

$$p(X_1 = -1.85, X_2 = 0.5, X_3 = -1.3, X_4 = 0.2)$$



Tractable inference

- ***Structural properties*** enable tractable inference of different queries

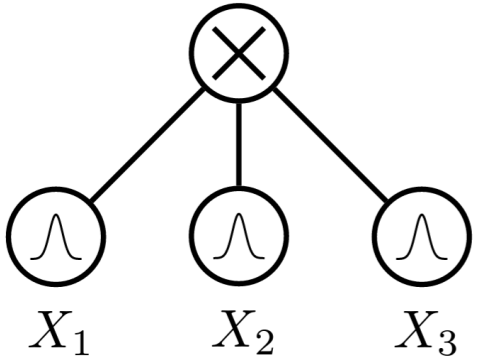
  polytime in the size of circuit (# edges)

- E.g., *marginal & conditional inference* takes *linear time* in the circuit size for *smooth* and *decomposable* PCs

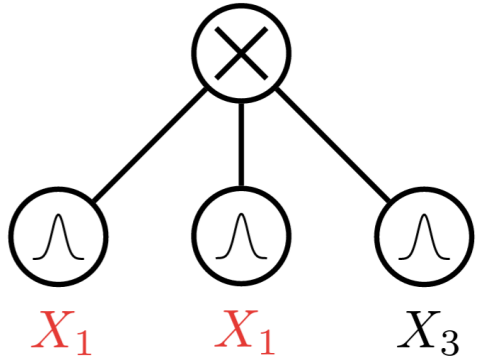
"What's the diagnosis if some test results are missing?"

Decomposability

- A product node is *decomposable* if its children are defined on disjoint sets of variables (i.e. their scopes are mutually exclusive)



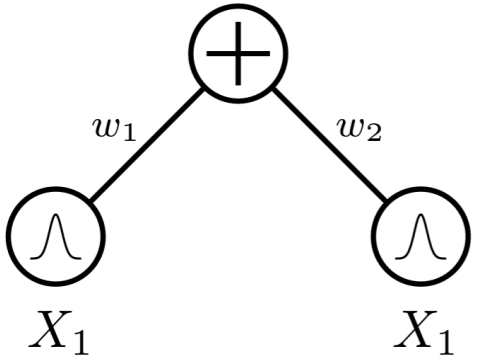
decomposable circuit



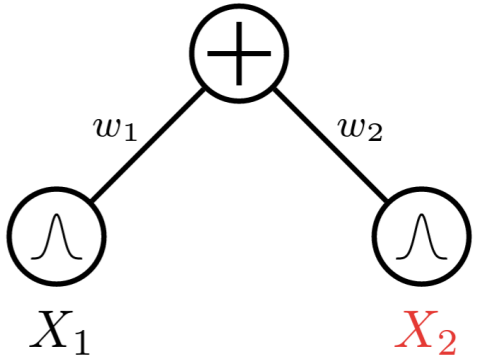
non-decomposable circuit

Smoothness

- A sum node is *smooth* if its children are defined on the same variables



smooth circuit

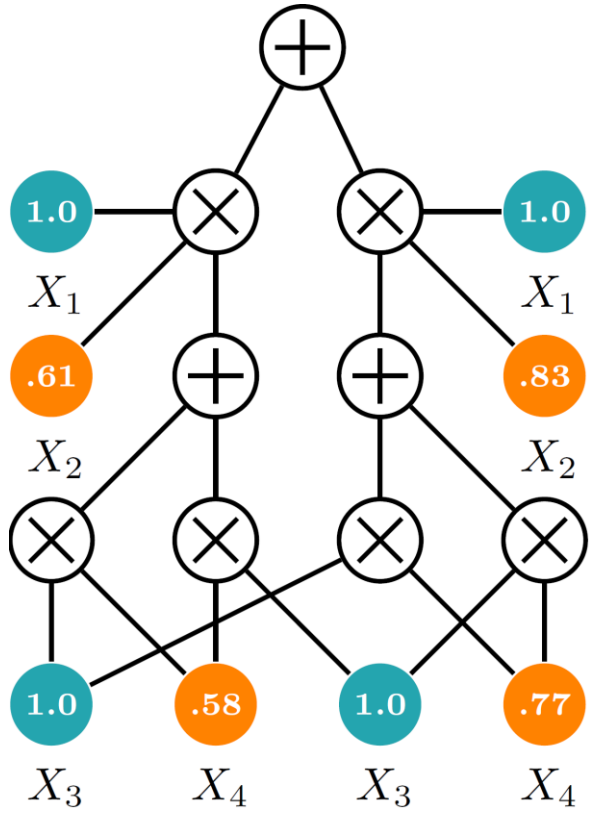


non-smooth circuit

Tractable marginals

smooth and ***decomposable*** PC =>
compute any marginal by a single evaluation

e.g. $p(x_2, x_4) = \int p(x_1, x_2, x_3, x_4)\, dX_1 X_3$



Outline

1. Probabilistic circuits – How structure enables tractable computation
2. A compositional view of inference – How tractable inference algorithms can be derived systematically
3. Tractable reasoning for distributional robustness

Given a new inference task, e.g.,

$$-\sum_{\boldsymbol{x}} p(\boldsymbol{x}) \log p(\boldsymbol{x})$$

Entropy

$$\sum_{\boldsymbol{x}} p(\boldsymbol{x}) \log \frac{p(\boldsymbol{x})}{q(\boldsymbol{x})}$$

KL divergence

$$\max_{\boldsymbol{x}} \sum_{\boldsymbol{y} \sim \boldsymbol{e}} p(\boldsymbol{x}, \boldsymbol{y})$$

Marginal MAP inference

$$\sum_{\boldsymbol{x}} p(\boldsymbol{z}) p(\boldsymbol{y} | \boldsymbol{x}, \boldsymbol{z})$$

Causal inference
(backdoor adjustment)

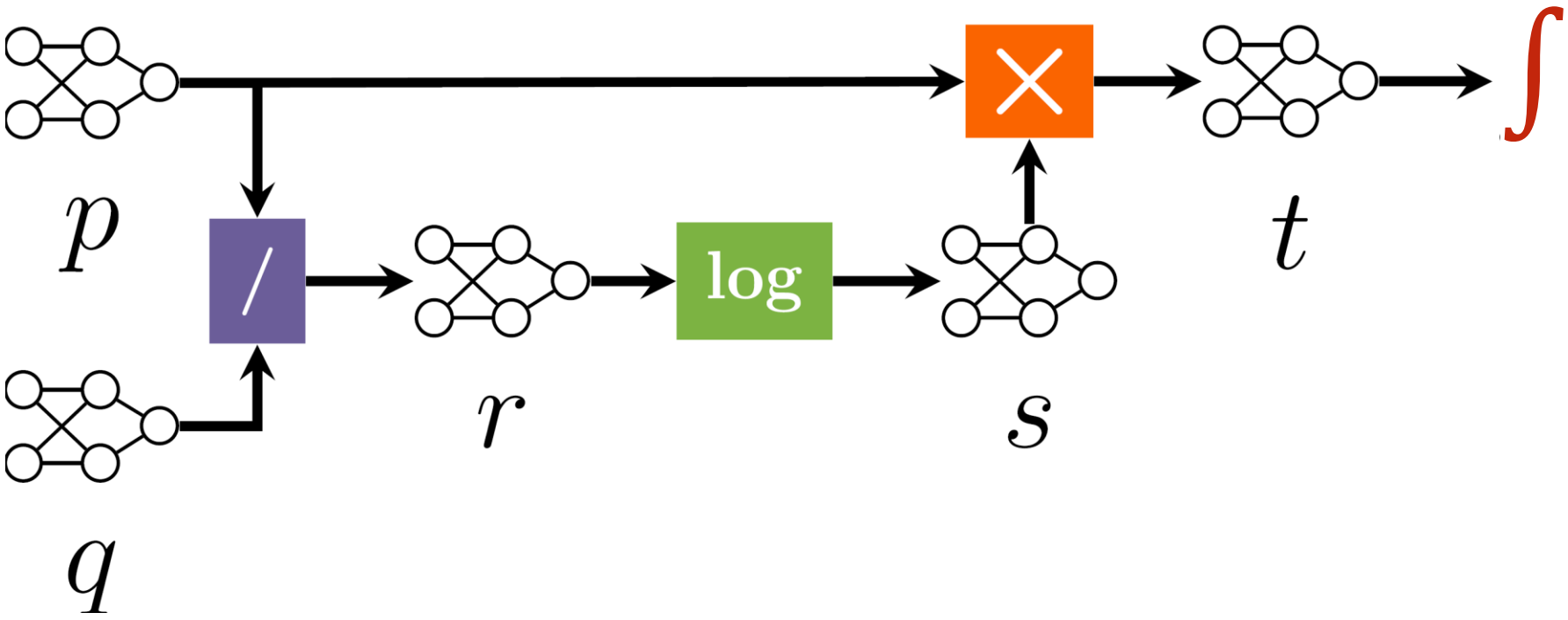
...and more...

Can we systematically derive efficient algorithms and circuit conditions for each query?

Inference as ***composition of operations!***

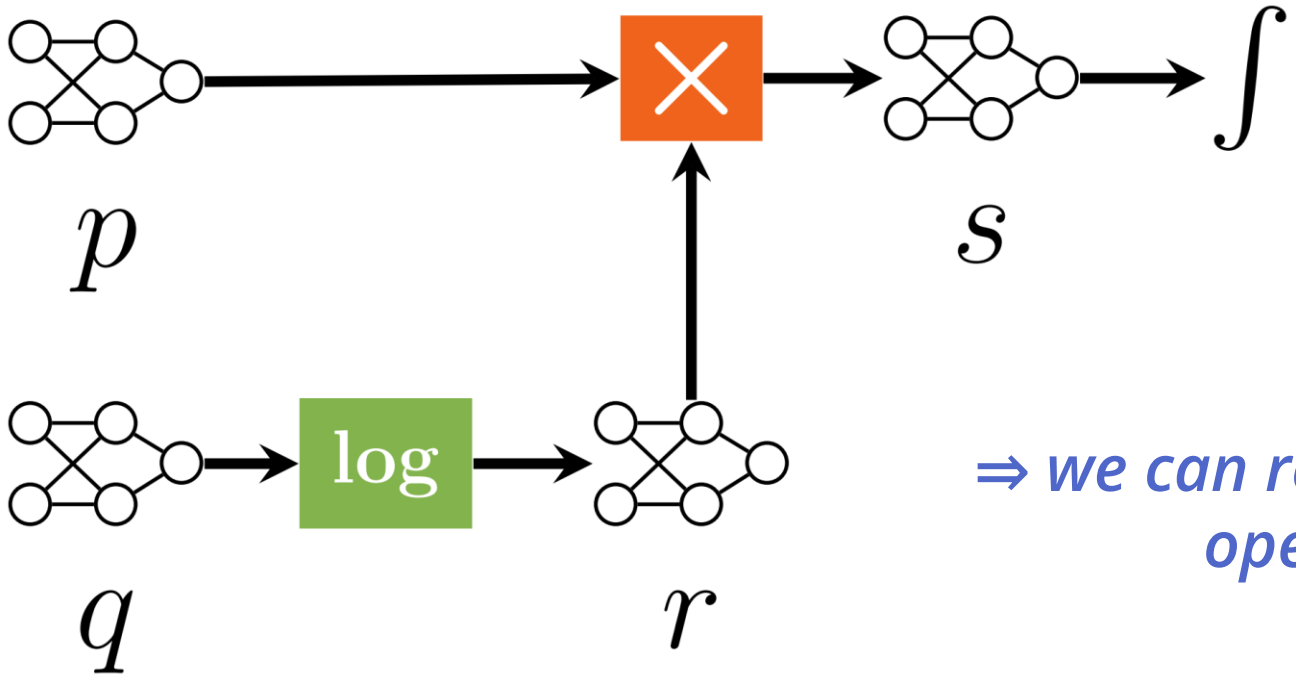
Queries as pipelines

$$\mathbb{KLD}(p \parallel q) = \int p(\boldsymbol{x}) \times \log((p(\boldsymbol{x})/q(\boldsymbol{x}))d\boldsymbol{X}$$



Queries as pipelines

$$H(p,q) = \int p(\boldsymbol{x}) \times \log(q(\boldsymbol{x}))d\boldsymbol{X}$$



⇒ *we can reuse the operations!*

Tractable circuit operations

Operation		Tractability		Hardness
		Input properties	Output properties	
SUM	$\theta_1 p + \theta_2 q$	(+Cmp)	(+SD)	NP-hard for Det output
PRODUCT	$p \cdot q$	Cmp (+Det, +SD)	Dec (+Det, +SD)	#P-hard w/o Cmp
POWER	$p^n, n \in \mathbb{N}$	SD (+Det)	SD (+Det)	#P-hard w/o SD
	$p^\alpha, \alpha \in \mathbb{R}$	Sm, Dec, Det (+SD)	Sm, Dec, Det (+SD)	#P-hard w/o Det
QUOTIENT	p/q	Cmp; q Det (+p Det,+SD)	Dec (+Det,+SD)	#P-hard w/o Det
LOG	$\log(p)$	Sm, Dec, Det	Sm, Dec	#P-hard w/o Det
EXP	$\exp(p)$	linear	SD	#P-hard

smooth, decomposable, deterministic

smooth, decomposable

Which conditions are needed…

…to efficiently perform the operation…

…while maintaining some circuit properties?

Inference by tractable operations

systematically derive tractable inference algorithm of complex queries

	Query	Tract. Conditions	Hardness
CROSS ENTROPY	$-\int p(\boldsymbol{x}) \log q(\boldsymbol{x}) \, \mathrm{d}\mathbf{X}$	Cmp, q Det	#P-hard w/o Det
SHANNON ENTROPY	$-\sum p(\boldsymbol{x}) \log p(\boldsymbol{x})$	Sm, Dec, Det	coNP-hard w/o Det
RÉNYI ENTROPY	$(1-\alpha)^{-1} \log \int p^{\alpha}(\boldsymbol{x}) \, d\mathbf{X}, \alpha \in \mathbb{N}$	SD	#P-hard w/o SD
	$(1-\alpha)^{-1} \log \int p^{\alpha}(\boldsymbol{x}) \, d\mathbf{X}, \alpha \in \mathbb{R}_{+}$	Sm, Dec, Det	#P-hard w/o Det
MUTUAL INFORMATION	$\int p(\boldsymbol{x}, \boldsymbol{y}) \log(p(\boldsymbol{x}, \boldsymbol{y})/(p(\boldsymbol{x})p(\boldsymbol{y})))$	Sm, SD, Det*	coNP-hard w/o SD
KULLBACK-LEIBLER DIV.	$\int p(\boldsymbol{x}) \log(p(\boldsymbol{x})/q(\boldsymbol{x})) d\mathbf{X}$	Cmp, Det	#P-hard w/o Det
RÉNYI'S ALPHA DIV.	$(1-\alpha)^{-1} \log \int p^{\alpha}(\boldsymbol{x}) q^{1-\alpha}(\boldsymbol{x}) \, d\mathbf{X}, \alpha \in \mathbb{N}$	Cmp, q Det	#P-hard w/o Det
	$(1-\alpha)^{-1} \log \int p^{\alpha}(\boldsymbol{x}) q^{1-\alpha}(\boldsymbol{x}) \, d\mathbf{X}, \alpha \in \mathbb{R}$	Cmp, Det	#P-hard w/o Det
ITAKURA-SAITO DIV.	$\int [p(\boldsymbol{x})/q(\boldsymbol{x}) - \log(p(\boldsymbol{x})/q(\boldsymbol{x})) - 1] d\,\mathbf{X}$	Cmp, Det	#P-hard w/o Det
CAUCHY-SCHWARZ DIV.	$-\log \frac{\int p(\boldsymbol{x}) q(\boldsymbol{x}) d\mathbf{X}}{\sqrt{\int p^2(\boldsymbol{x}) d\mathbf{X} \int q^2(\boldsymbol{x}) d\mathbf{X}}}$	Cmp	#P-hard w/o Cmp
SQUARED LOSS	$\int (p(\boldsymbol{x}) - q(\boldsymbol{x}))^2 d\,\mathbf{X}$	Cmp	#P-hard w/o Cmp

Compositional inference algorithms

```
function kld(p, q)
    r = quotient(p, q)
    s = log(r)
    t = product(p, s)
    return integrate(t)
end
```

```
function xent(p, q)
    r = log(q)
    s = product(p, r)
    return -integrate(s)
end
```

```
function ent(p)
    q = log(p)
    r = product(p, q)
    return -integrate(s)
end
```

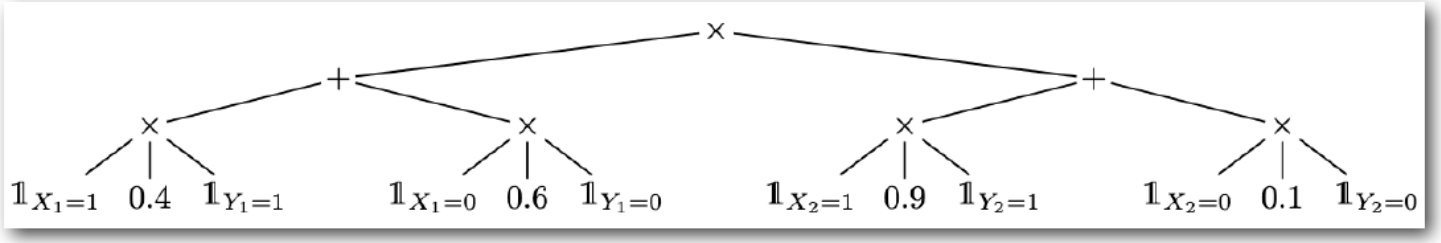
```
function alphadiv(p, q, alpha=1.5)
    r = real_pow(p, alpha)
    s = real_pow(q, 1.0-alpha)
    t = product(r, s)
    return log(integrate(t)) / (1.0-alpha)
end
```

Efficient inference algorithms ***in a couple lines of Julia code!*** [1]

[1] https://github.com/UCLA-StarAI/circuit-ops-atlas

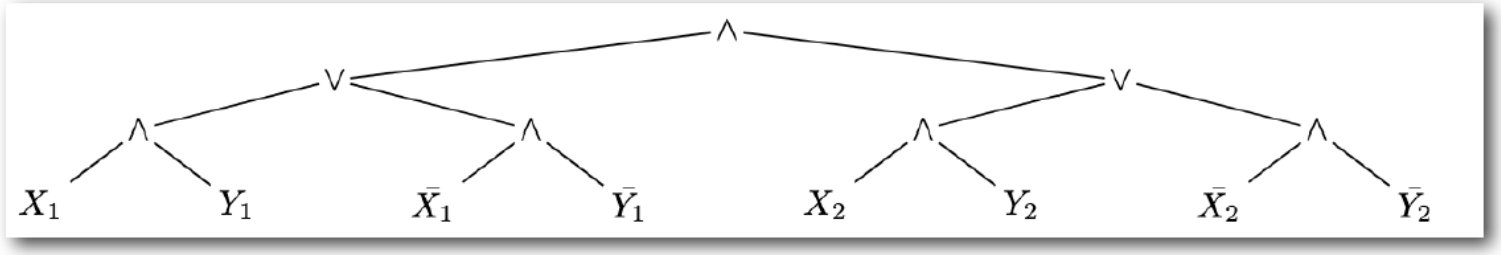
Algebraic circuits

Let $\mathcal{S} = (S, \oplus, \otimes, 0_{\mathcal{S}}, 1_{\mathcal{S}})$ be a commutative semiring. An algebraic circuit C over $\mathcal{S}$ recursively defines $f: \text{Assign}(\mathbf{X}) \rightarrow S$ as a computation graph of $\oplus$ and $\otimes$



Probabilistic Semiring
$(\mathbb{R}_{\geq 0}, +, \times, 0, 1)$

probabilistic circuits: PC, SPN, PSDD, Cutset Networks, etc.



Boolean Semiring
$(\{\bot\ \top\}, \vee, \wedge, \bot, \top)$

logical circuits: BDD, DNNF, DFA, etc.

Algebraic queries…

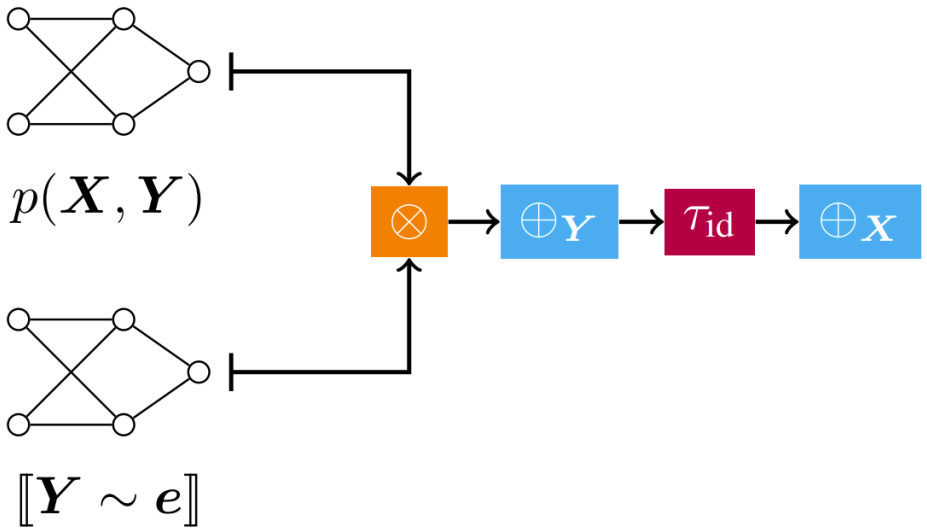
$$\max_{\boldsymbol{x}} \sum_{\boldsymbol{y}\sim\boldsymbol{e}} p(\boldsymbol{x},\boldsymbol{y})$$

Marginal MAP
for a PC p(X,Y)

Anatomy of MMAP:

- $\boldsymbol{y}\sim\boldsymbol{e}$: **product** with logical circuit $[\![\boldsymbol{Y}\sim\boldsymbol{e}]\!]$
- $\sum_{\boldsymbol{y}}$: **summation** over semiring $(\mathbb{R}_{\geq 0}, +, \times, 0, 1)$
- $\max_{\boldsymbol{x}}$: **summation** over semiring $(\mathbb{R}_{\geq 0}, \max, \times, 0, 1)$

…as composition of operations…

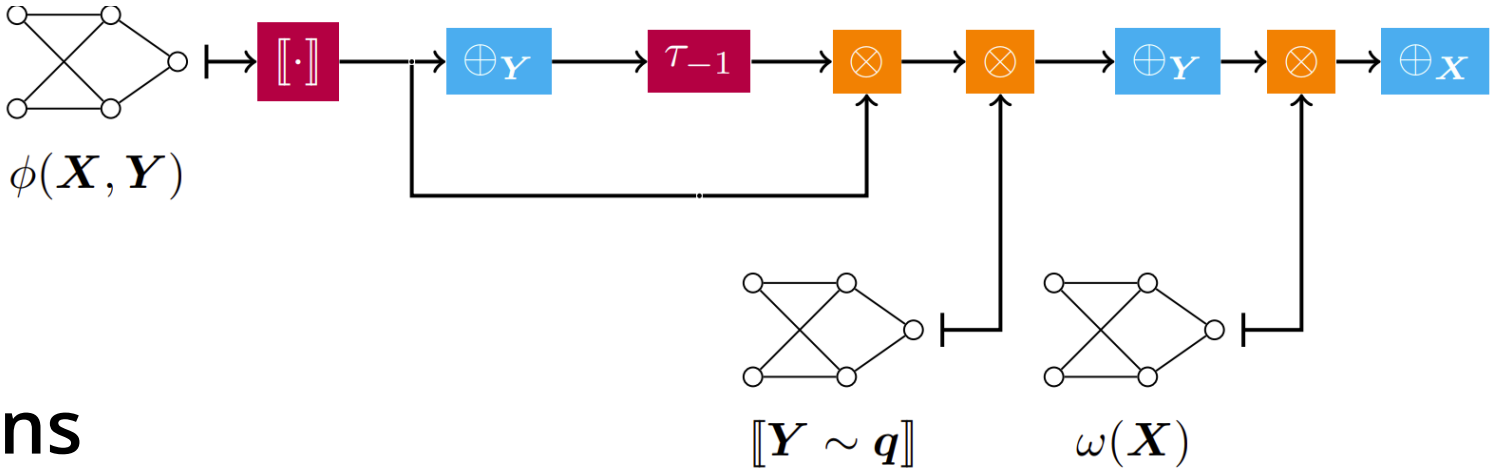


Algebraic queries…

$$\sum_{\boldsymbol{x}} w(\boldsymbol{x}) \sum_{\boldsymbol{y} \sim \boldsymbol{q}} \frac{\phi(\boldsymbol{x}, \boldsymbol{y})}{\sum_{\boldsymbol{y}'} \phi(\boldsymbol{x}, \boldsymbol{y}')}$$

Probabilistic answer set programming inference
(maximum entropy semantics)

…as composition of operations…



…and reuse operations

2AMC [Kiesel et al., 2022]

Class of inference problems
inc. MMAP, PLP inference, SDP, etc.

$$\bigoplus_{\boldsymbol{x}}\left(\tau_{\mathcal{S}_Y\to\mathcal{S}_X}\left(\bigoplus_{\boldsymbol{y}}[\![\phi(\boldsymbol{x},\boldsymbol{y})]\!]_{\mathcal{B}\to\mathcal{S}_Y}\otimes\omega(\boldsymbol{y})\right)\otimes\omega'(\boldsymbol{x})\right)$$

Corollary 2AMC can be computed in $O(|\phi|)$ for deterministic, **X**-deterministic, and **X**-first circuits ϕ

Further, we can relax these conditions for specific 2AMC problems.

Causal inference

Adjustment formulae

$$p(\boldsymbol{y}|do(\boldsymbol{x}))=\sum_{\boldsymbol{z}}p(\boldsymbol{z})p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{z}),$$

$$p(\boldsymbol{y}|do(\boldsymbol{x}))=\sum_{\boldsymbol{z}}p(\boldsymbol{z}|\boldsymbol{x})\sum_{\boldsymbol{x}'}p(\boldsymbol{x}')p(\boldsymbol{y}|\boldsymbol{x}',\boldsymbol{z}).$$

Corollary Backdoor adjustment can be computed in $O(|p|)$ for a **Z**-det and $(\mathbf{X}\cup\mathbf{Z})$-det circuit p

Corollary Frontdoor adjustment can be computed in $O(|p|^2)$ for a **X**-det and $(\mathbf{X}\cup\mathbf{Z})$-det circuit

Improves on previously best-known *custom* algorithms!

Outline

1. Probabilistic circuits – How structure enables tractable computation
2. A compositional view of inference – How tractable inference algorithms can be derived systematically
3. Tractable reasoning for distributional robustness

Robust Maximum-Likelihood Learning

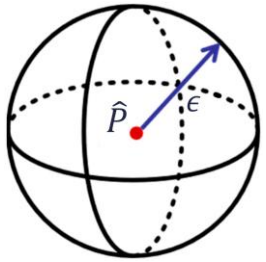
Maximum-likelihood estimation: given a set of samples $\{\boldsymbol{x}^i\}_{i=1,\dots,N}$ forming an empirical distribution $\hat{P}(\boldsymbol{X})$, learn the model P that maximizes (log-)likelihood

$$\max_P \mathbb{E}_{\hat{P}}\left[\log P(\boldsymbol{X})\right]$$

However, often *not reliable* under *data noise/perturbations* or *distribution shifts*

Robust Maximum-likelihood learning: maximize likelihood wrt the *worst-case distribution* Q within ϵ-*Wasserstein ball* of $\hat{P}$

$$\max_P \min_Q \mathbb{E}_Q\left[\log P(\mathbf{X})\right]$$
$$s.t.\, \mathsf{W}(\hat{P}, Q) \leq \epsilon$$

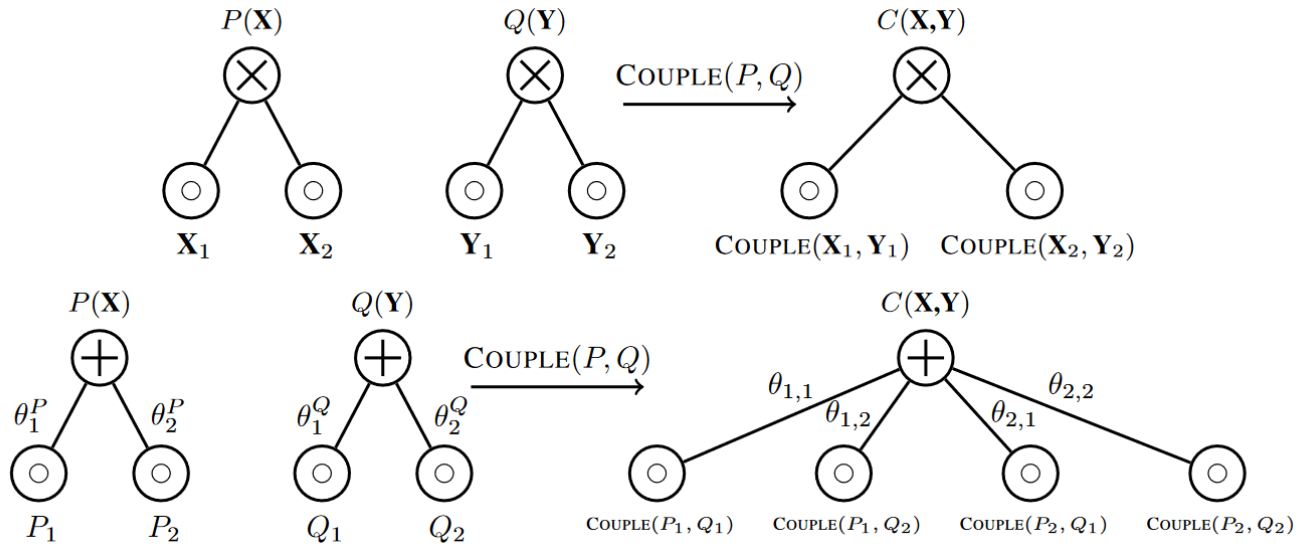


Circuit-Wasserstein Distance

$$\mathsf{W}_p^p(P,Q) = \inf_{\gamma \in \Gamma(P,Q)} \mathbb{E}_{\gamma(\mathbf{x},\mathbf{y})}[\|\mathbf{x}-\mathbf{y}\|_p^p]$$

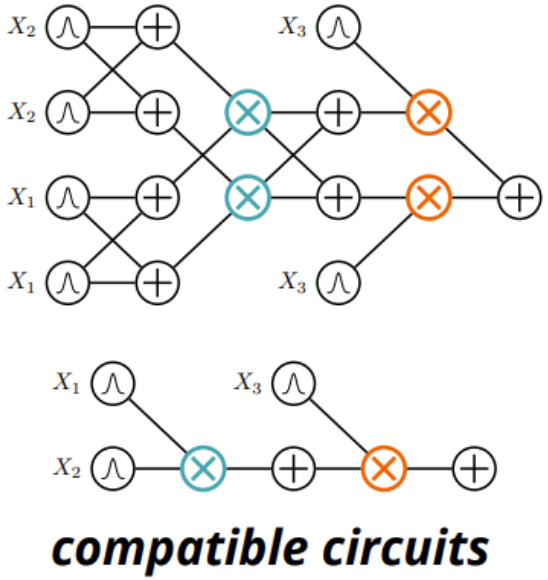
If *P* and *Q* are PCs with ***compatible structure,*** we can construct *coupling circuits*,

and minimize the Wasserstein objective in ***quadratic time*** (solving a series of small LPs)

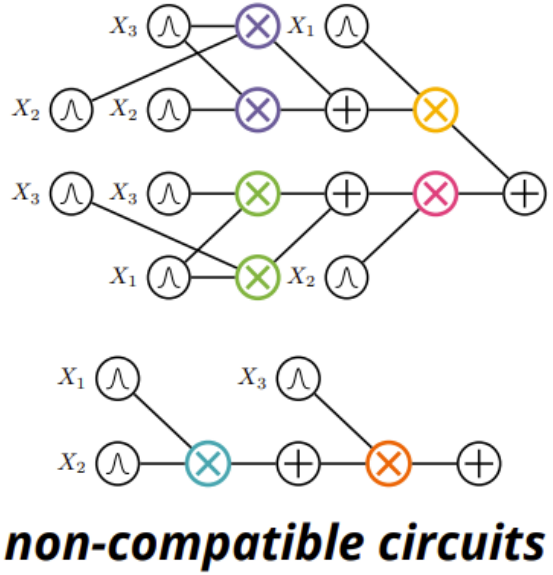


Compatibility

Two circuits are *compatible* if they have the same *hierarchical scope partitioning*



compatible circuits



non-compatible circuits

support polytime products

PeTeR: Post-training Robustification of PCs

Given a *pre-trained* PC $\hat{P}$, learn the PC parameters P_θ that guarantees high likelihood on the worst-case distribution Q_φ within ϵ-Circuit-Wasserstein ball of $\hat{P}$.

$$\max_{P_\theta} \min_{Q_\phi} \mathbb{E}_{Q_\phi}[\log P_\theta(\mathbf{X})]$$
$$\text{s.t. } \mathsf{CW}(\hat{P}, Q_\phi) \leq \epsilon$$

Note: updates a pre-trained PC. No training data needed!

Solve the equivalent Lagrangian problem by gradient descent-ascent

Tractable *expectation*, *circuit-Wasserstein*, & *gradient* computation thanks to ***compatibility***!

adversarially-perturbed *randomly-perturbed*

		$\mathcal{T}$			$\mathcal{T}_a$			$\mathcal{T}_r$		
Dataset	h	MLE-PC	RL-TPM	**PeTeR**	MLE-PC	RL-TPM	**PeTeR**	MLE-PC	RL-TPM	**PeTeR**
NLTCS	1	**-6.09**	-9.36	-6.79	-11.26	-10.91	**-9.31**	-8.41 ±0.03	-10.13 ±0.03	**-8.04** ±0.02
	3		-10.34	-7.79	-18.43	**-11.14**	-11.56	-11.46 ±0.05	-11.22 ±0.02	**-9.79** ±0.02
	5		-10.86	-9.90	-23.02	**-10.61**	-11.75	-13.24 ±0.09	-11.44 ±0.02	**-10.76** ±0.01
MSNBC	1	**-6.43**	-9.14	-7.05	-12.01	-10.33	**-8.45**	-8.31 ±0.01	-9.88 ±0.01	**-8.13** ±0.00
	3		-8.47	-8.69	-19.88	-11.52	**-10.96**	-11.29 ±0.01	-10.17 ±0.01	**-9.99** ±0.00
	5		-9.92	-10.76	-25.39	**-11.85**	-11.95	-13.52 ±0.02	**-10.97** ±0.00	-11.29 ±0.00
Plants	1	**-14.46**	-21.01	-14.98	-23.39	-22.59	**-19.62**	-18.78 ±0.03	-23.65 ±0.04	**-18.03** ±0.02
	3		-22.30	-16.16	-38.81	**-23.22**	-26.69	-26.26 ±0.09	-28.76 ±0.05	**-23.31** ±0.05
	5		-23.64	-17.23	-52.40	**-27.54**	-33.30	-32.36 ±0.11	-32.63 ±0.04	**-27.49** ±0.07
Netflix	1	**-57.59**	-58.82	-57.66	-61.13	-60.67	**-60.61**	**-58.27** ±0.02	-59.37 ±0.01	-58.32 ±0.02
	3		-59.27	-58.09	-67.12	**-63.08**	-65.06	**-59.56** ±0.03	-60.55 ±0.02	-59.69 ±0.03
	5		-60.10	-59.21	-72.17	**-65.32**	-66.48	**-60.78** ±0.06	-61.88 ±0.04	-61.28 ±0.04
DNA	1	**-79.97**	-82.69	-81.05	-87.68	-86.67	**-85.12**	-83.44 ±0.09	-84.65 ±0.05	**-83.42** ±0.05
	3		-84.31	-82.62	-102.88	-93.38	**-90.70**	-90.33 ±0.15	-88.80 ±0.05	**-88.07** ±0.07
	5		-85.85	-84.23	-117.82	-98.24	**-96.08**	-97.00 ±0.23	-92.00 ±0.08	**-91.18** ±0.10
Movie	1	**-52.81**	-55.02	-54.07	-61.93	-61.21	**-58.59**	**-58.53** ±0.09	-59.77 ±0.07	-58.99 ±0.08
	3		-57.06	-53.38	-79.51	-69.61	**-67.62**	-69.77 ±0.20	**-68.82** ±0.12	-68.91 ±0.17
	5		-59.34	-56.09	-96.75	**-78.24**	-79.51	-80.66 ±0.16	-77.02 ±0.11	**-75.07** ±0.16
BBC	1	**-250.20**	-250.72	-250.30	-258.72	**-254.99**	-256.75	**-253.04** ±0.12	-253.43 ±0.10	-253.10 ±0.11
	3		-251.45	-250.55	-275.07	**-262.65**	-266.11	**-258.74** ±0.15	-259.25 ±0.12	-258.82 ±0.13
	5		-252.52	-250.58	-290.24	**-269.24**	-274.58	-264.30 ±0.30	-265.01 ±0.26	**-264.16** ±0.29

PeTeR improves the performance of pre-trained PCs on adversarially- and randomly-perturbed datasets, and frequently outperforms baseline robust learning method

Ciotinga, Dai, and Choi, “PETER: Post-Training Robustification of Probabilistic Circuits”, 2026
Peddi, Rahman, and Gogate, “Robust learning of tractable probabilistic models”, UAI 2022

Takeaways

1. Probabilistic circuits – How structure enables tractable computation
2. A compositional view of inference – How tractable inference algorithms can be derived systematically
3. Tractable reasoning for distributional robustness

What was not in this talk:

- How to build/learn the circuits
- Expressive efficiency vs tractability trade-off (for both exact & approximate)

Thank you!

Thanks to my students and collaborators!