UCLA Computer Science

STAR AI
RESEARCH LAB
UCLA

Tractable Probabilistic Circuits

Guy Van den Broeck

🦊 FOCS 2023 Workshop: Algorithmic Aspects of High-Dimensional Probabilistic Models - Oct 3 2023

Overview

1. What are probabilistic circuits?

   tractable deep generative models

2. What are they useful for?

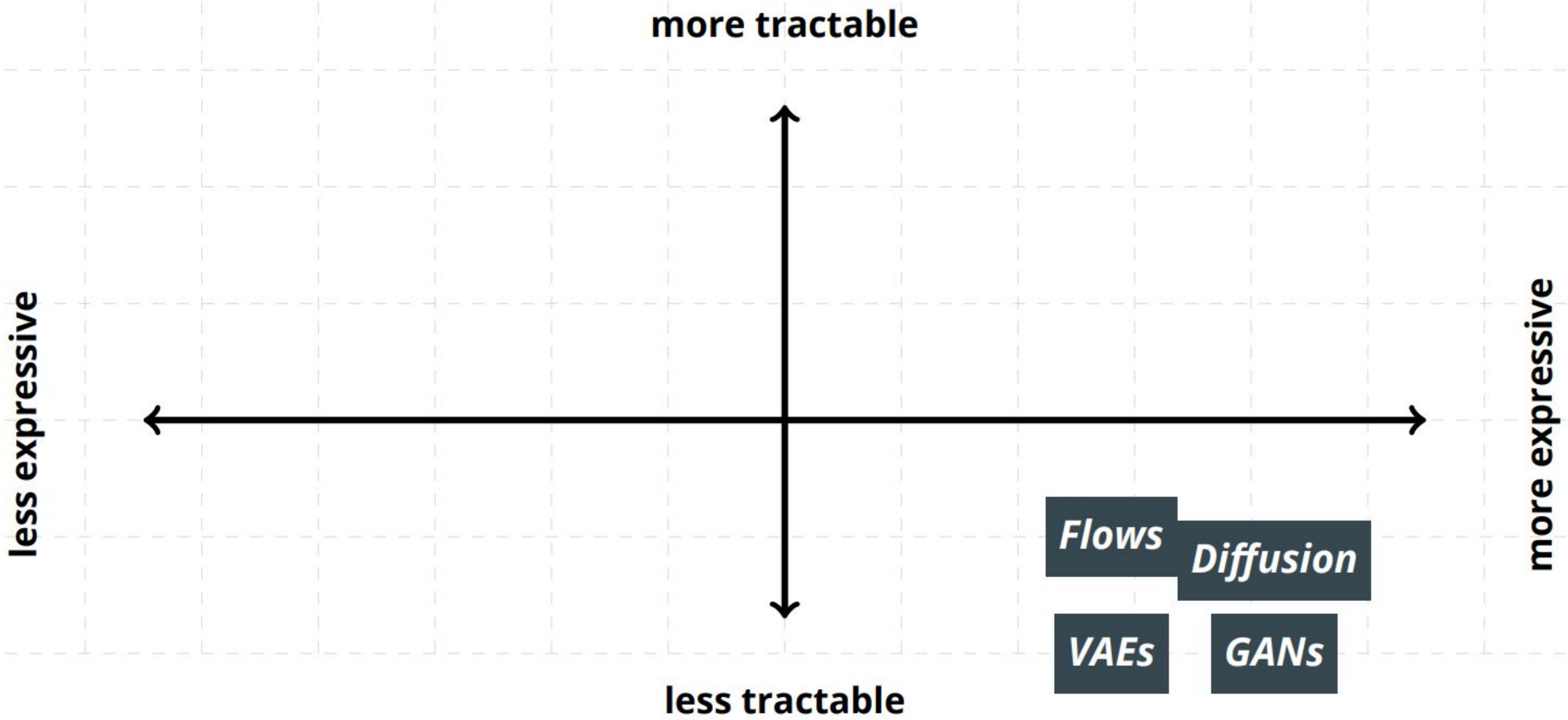
   controlling generative AI

3. What is the underlying theory?

   probability generating polynomials

$$\Pr(X) = \sum_Y \Pr(X, Y)$$

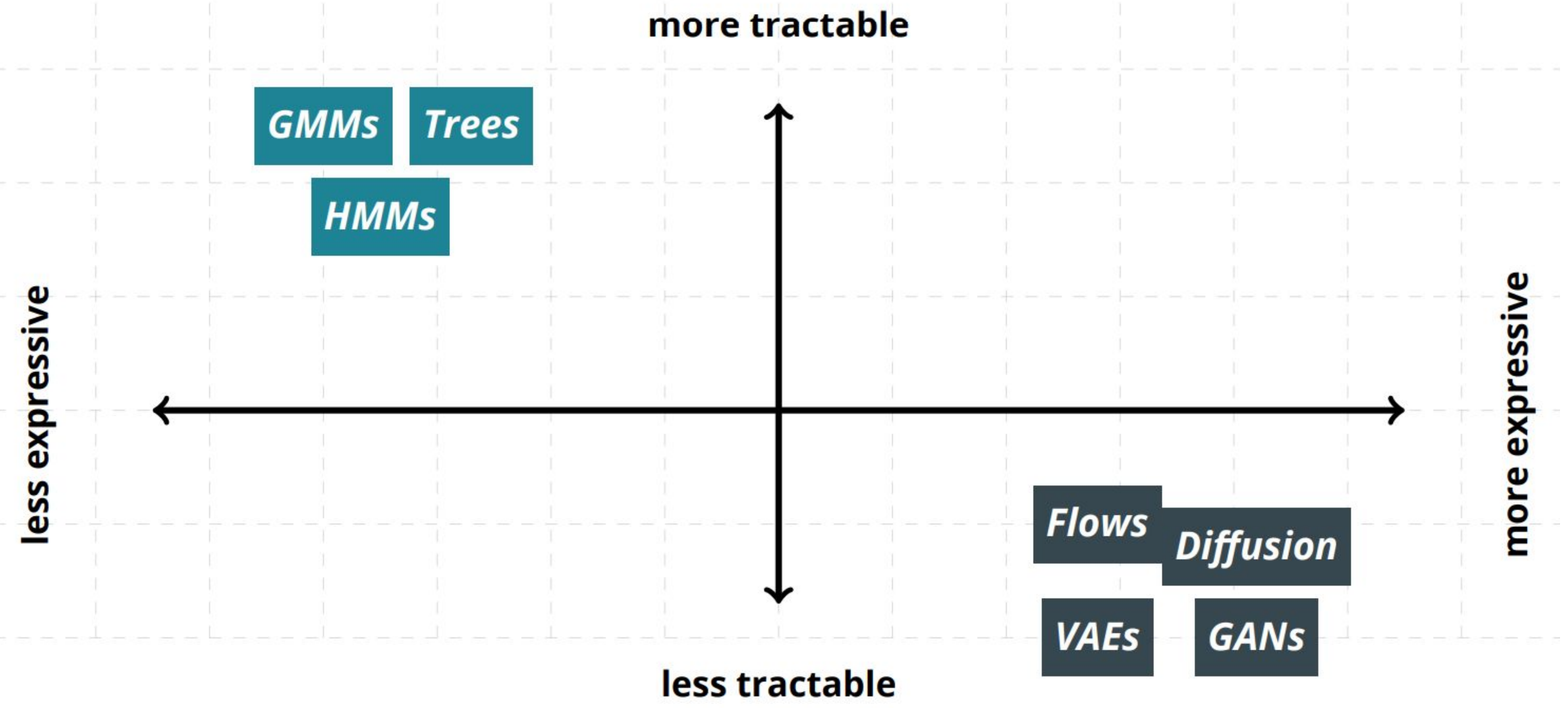
🦊 High-dimensional probabilistic models take various forms: classically-studied models such as multivariate Gaussians and Erdős-Rényi graphs, models with roots in statistical physics such as stochastic block models and Ising models, probabilistic graphical models such as Bayesian networks and Markov random fields, as well as the class of implicit generative models, such as generative adversarial networks and large language models 🦊

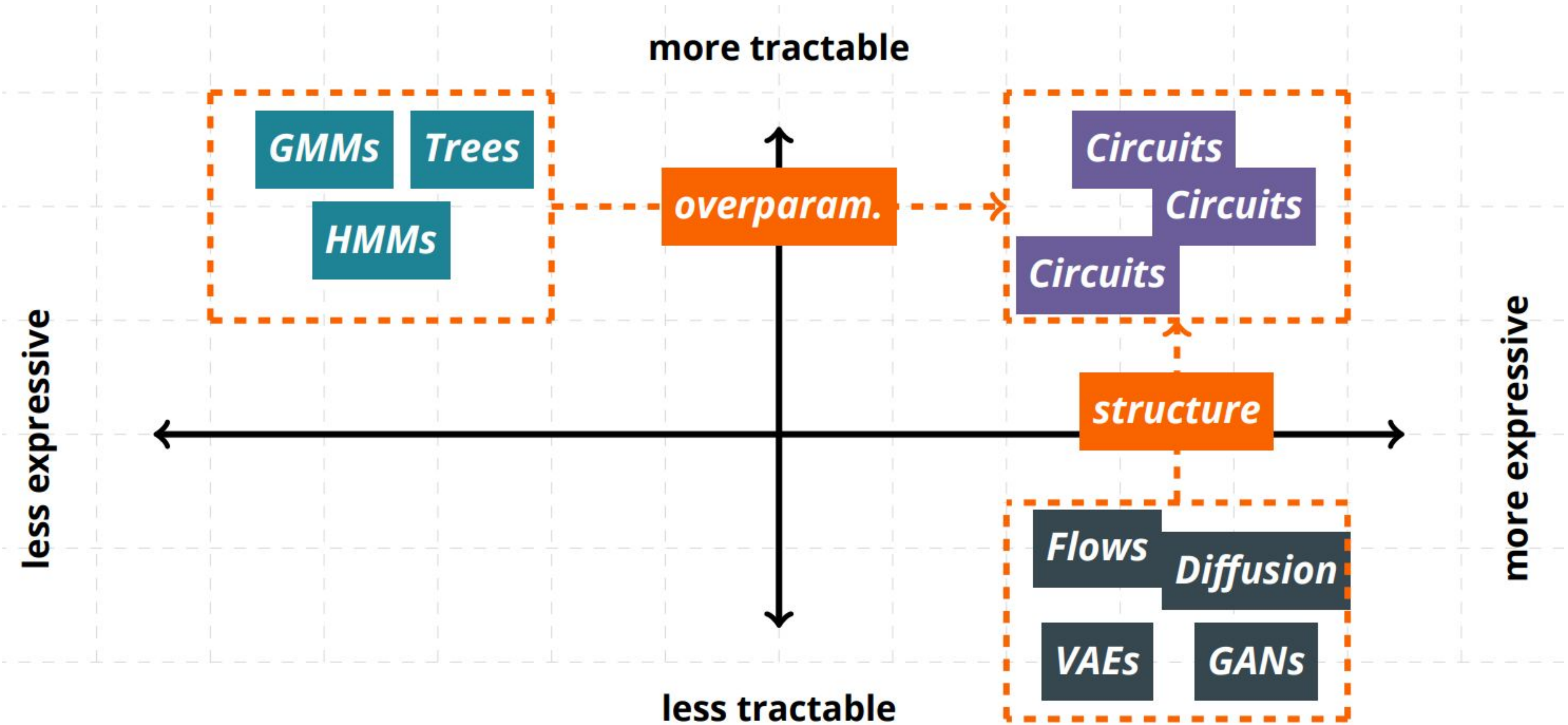


more tractable
less expressive
more expressive
Flows
Diffusion
VAEs
GANs
less tractable

$$\Pr(X) = \sum_Y \Pr(X, Y)$$

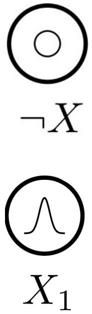
🦊 High-dimensional probabilistic models take various forms: classically-studied models such as multivariate Gaussians and Erdős-Rényi graphs, models with roots in statistical physics such as stochastic block models and Ising models, probabilistic graphical models such as Bayesian networks and Markov random fields, as well as the class of implicit generative models, such as generative adversarial networks and large language models 🦊


more tractable
GMMs
Trees
HMMs
less expressive
more expressive
Flows
Diffusion
VAEs
GANs
less tractable


more tractable
GMMs
Trees
HMMs
overparam.
Circuits
Circuits
Circuits
less expressive
structure
more expressive
Flows
Diffusion
VAEs
GANs
less tractable

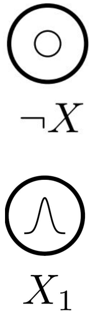
Probabilistic circuits

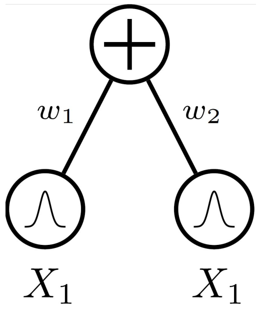
computational graphs that recursively define distributions

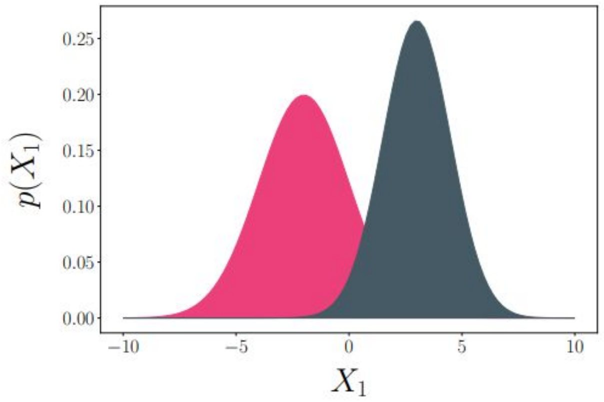


Probabilistic circuits

computational graphs that recursively define distributions







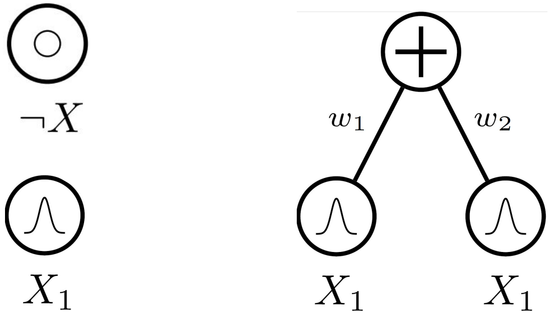
$$p(X_1) = w_1 p_1(X_1) + w_2 p_2(X_1)$$

$\Rightarrow$ *mixtures*

$$p(X) = p(Z = 1) \cdot p_1(X|Z = 1) + p(Z = 2) \cdot p_2(X|Z = 2)$$

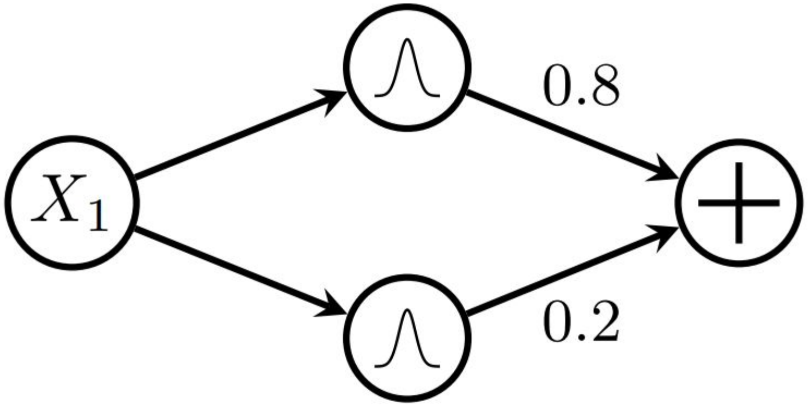
Probabilistic circuits

computational graphs that recursively define distributions



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

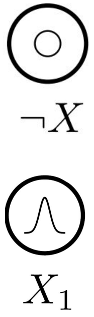
⇒ *mixtures*

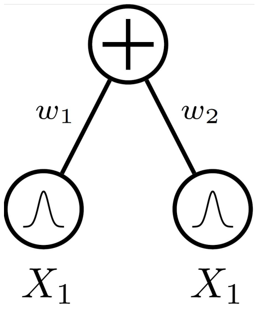


⇒ *if you prefer arithmetic circuit syntax with one single input X_1*

Probabilistic circuits

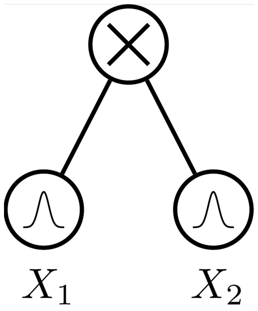
computational graphs that recursively define distributions





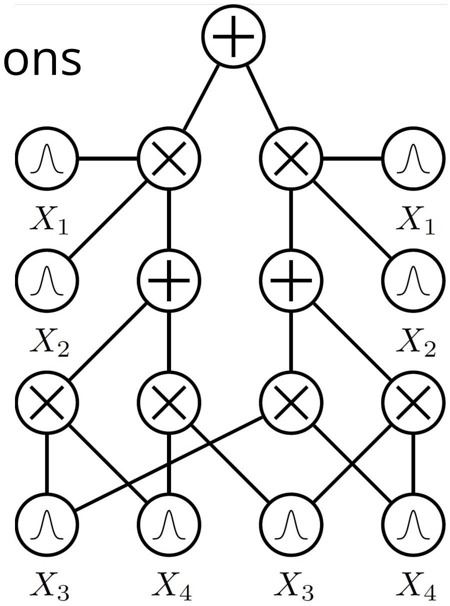
$p(X_1) = w_1 p_1(X_1) + w_2 p_2(X_1)$

⇒ *mixtures*



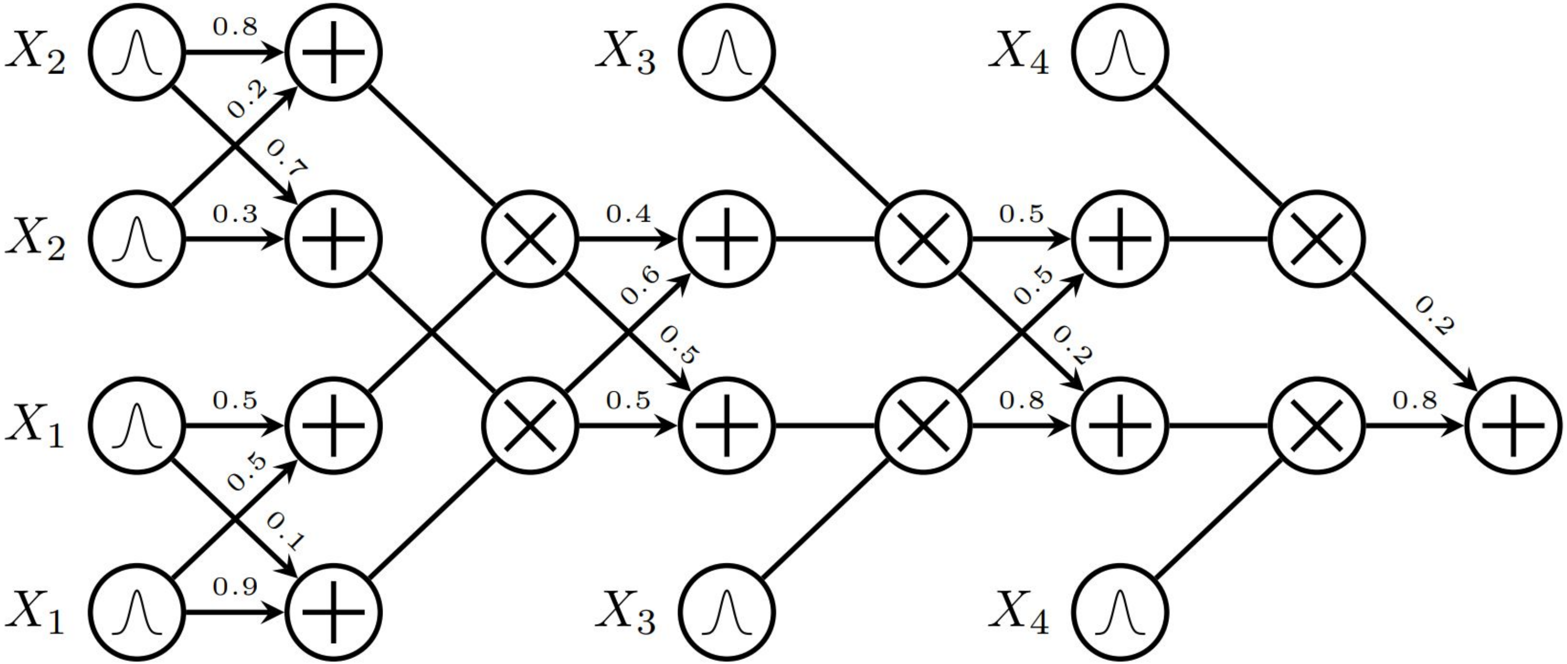
$p(X_1, X_2) = p(X_1) \cdot p(X_2)$

⇒ *factorizations*



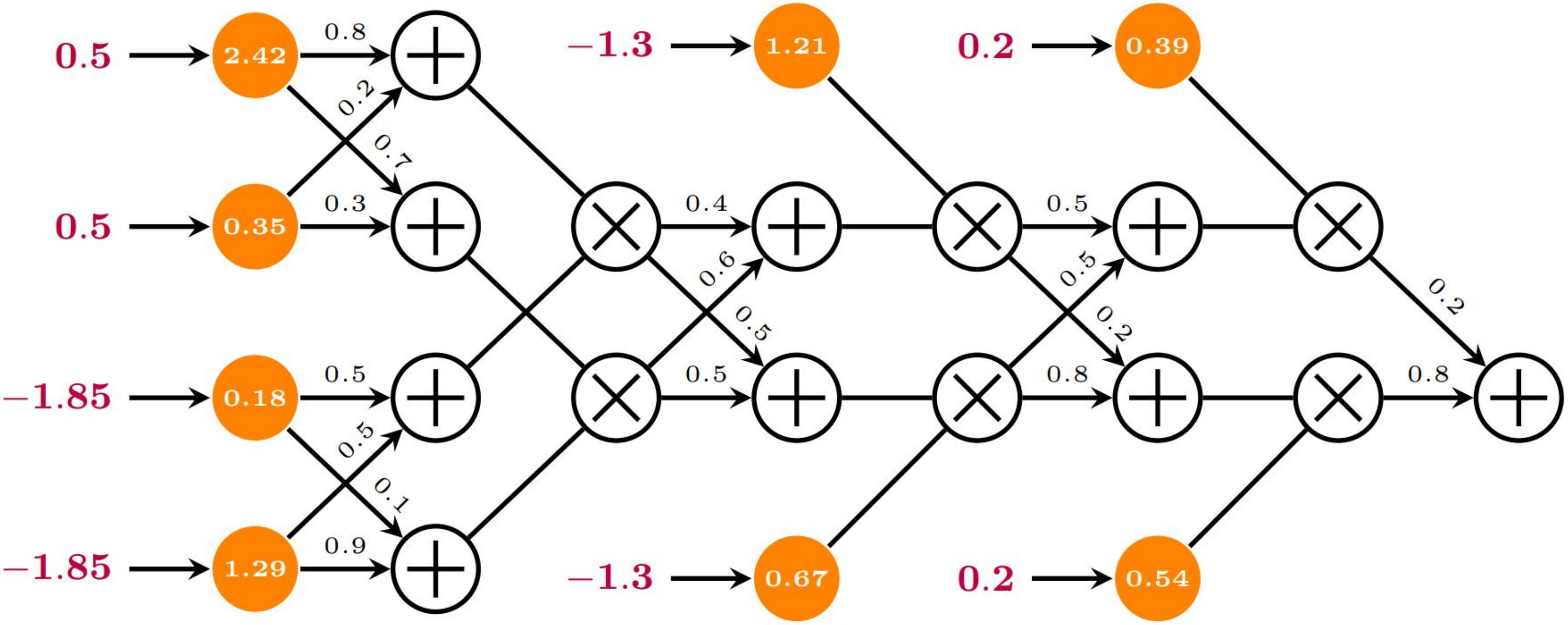
Likelihood

$p(X_1 = \mathbf{-1.85}, X_2 = \mathbf{0.5}, X_3 = \mathbf{-1.3}, X_4 = \mathbf{0.2})$



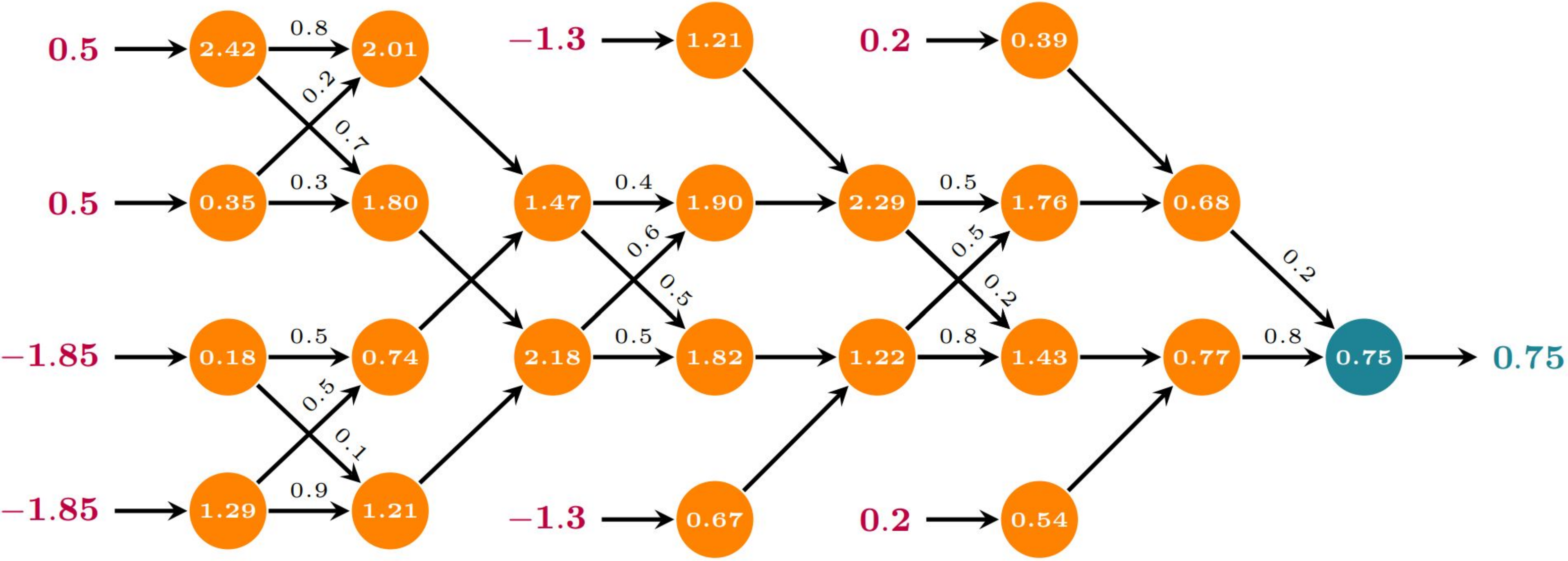
Likelihood

$$p(X_1 = \mathbf{-1.85}, X_2 = \mathbf{0.5}, X_3 = \mathbf{-1.3}, X_4 = \mathbf{0.2})$$



Likelihood

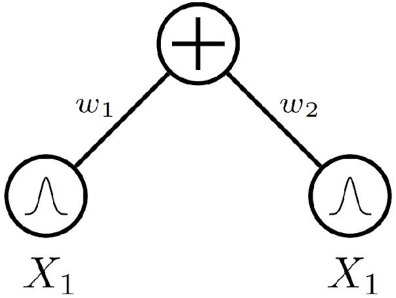
$$p(X_1 = \mathbf{-1.85}, X_2 = \mathbf{0.5}, X_3 = \mathbf{-1.3}, X_4 = \mathbf{0.2})$$



Tractable marginals

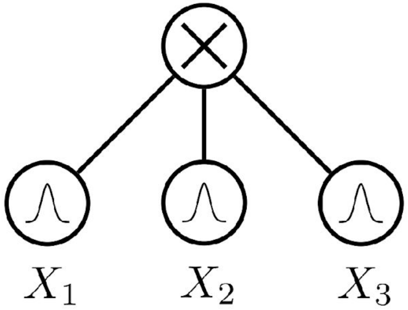
A sum node is ***smooth*** if its children depend on the same set of variables.

A product node is ***decomposable*** if its children depend on disjoint sets of variables.



smooth circuit

(~homogeneous)



decomposable circuit

(=syntactically multilinear)

Darwiche and Marquis, "A Knowledge Compilation Map", 2002

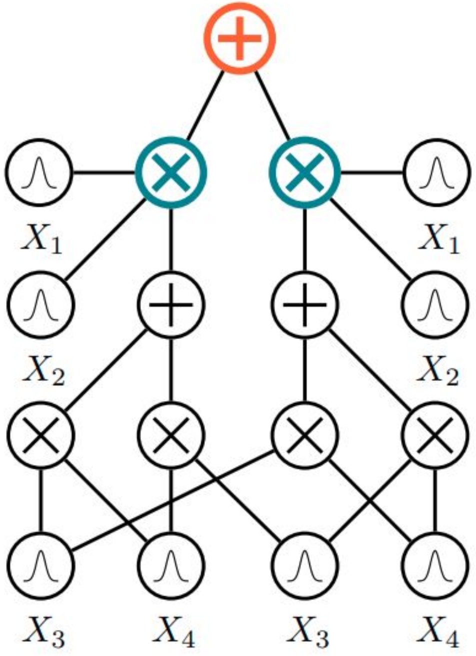
Smoothness + decomposability = tractable MAR

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

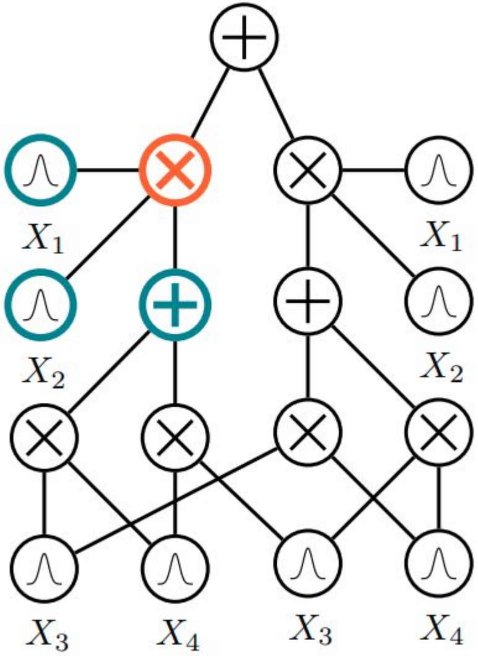
If $p(\mathbf{x}, \mathbf{y}, \mathbf{z}) = p(\mathbf{x})p(\mathbf{y})p(\mathbf{z})$, (***decomposability***):

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

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

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

$\Longrightarrow$ *integrals decompose into easier ones*



Smoothness + *decomposability* = *tractable MAR*

Forward pass evaluation for MAR

$\Rightarrow$ *linear in circuit size!*

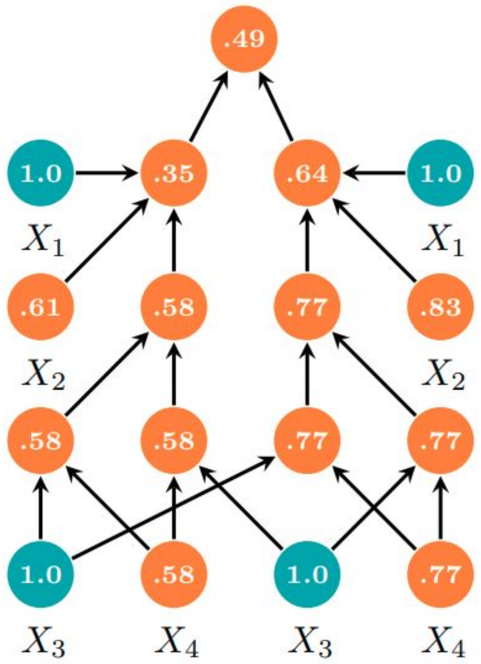
E.g. to compute $p(x_2, x_4)$:

- 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)



Generate a sentence using "frisbee", "caught" and "dog", following the given order.

Generate a sentence using "frisbee", "caught" and "dog", following the given order.

After a perfect throw, the frisbee glided through the air, and the dog, with incredible agility, caught it mid-flight.

ChatGPT

Generate a sentence using "frisbee", "caught" and "dog", following the given order.

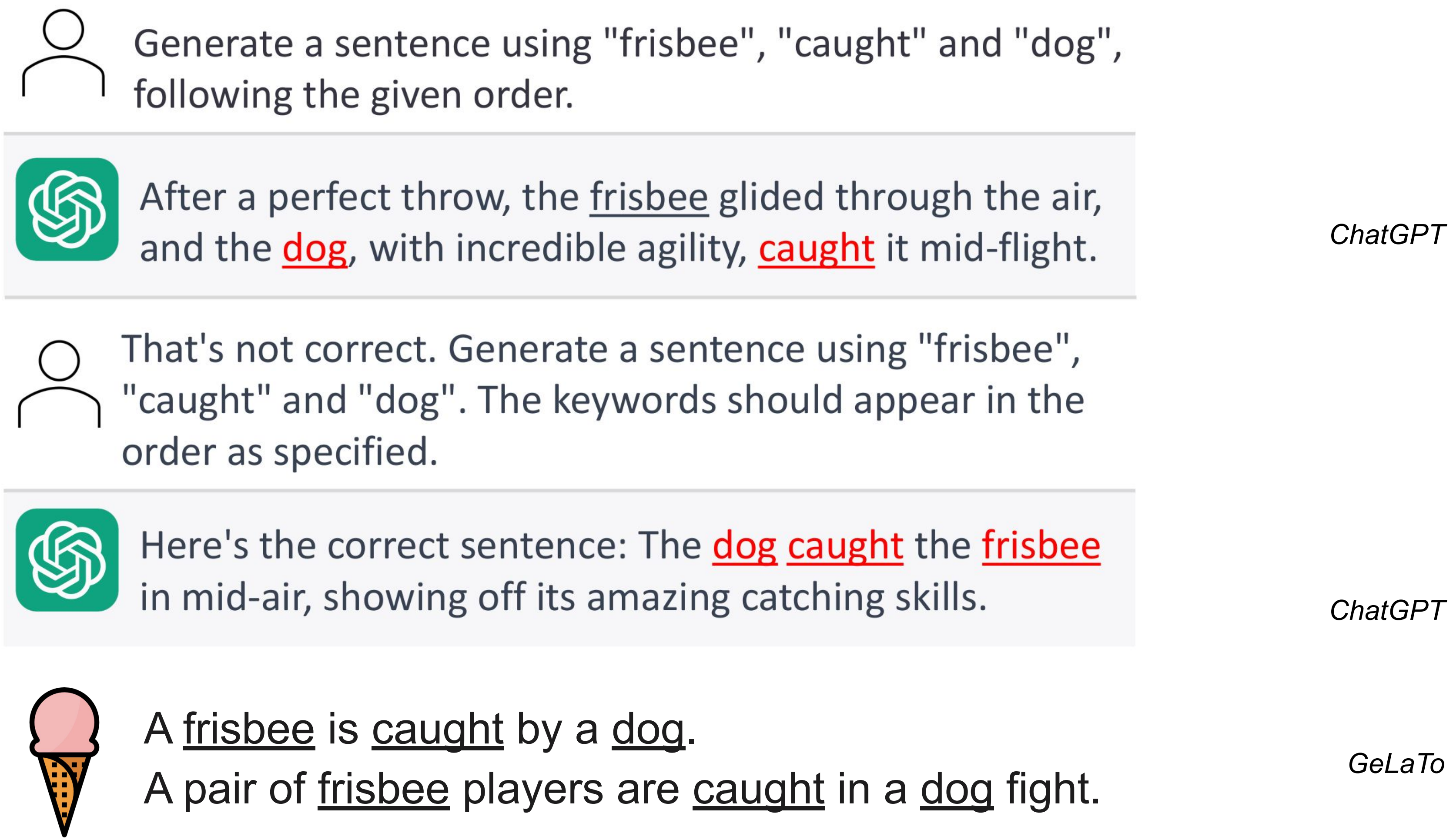
After a perfect throw, the frisbee glided through the air, and the dog, with incredible agility, caught it mid-flight.

ChatGPT

That's not correct. Generate a sentence using "frisbee", "caught" and "dog". The keywords should appear in the order as specified.

Here's the correct sentence: The dog caught the frisbee in mid-air, showing off its amazing catching skills.

ChatGPT


Generate a sentence using "frisbee", "caught" and "dog", following the given order.
After a perfect throw, the frisbee glided through the air, and the dog, with incredible agility, caught it mid-flight.
ChatGPT
That's not correct. Generate a sentence using "frisbee", "caught" and "dog". The keywords should appear in the order as specified.
Here's the correct sentence: The dog caught the frisbee in mid-air, showing off its amazing catching skills.
ChatGPT
A frisbee is caught by a dog.
A pair of frisbee players are caught in a dog fight.
GeLaTo

What do we need?

Prefix: "The weather is"

Constraint α: text contains "winter"

Model only does $p(\text{next-token}|\text{prefix}) =$

cold	0.05
warm	0.10

We need $p(\text{next-token}|\text{prefix}, \alpha) =$

cold	0.50
warm	0.01

$$\propto \sum_{\text{text}} p(\text{next-token}, \text{text}, \text{prefix}, \alpha)$$

Marginalization!

CommonGen: a Challenging Benchmark

Given 3-5 keywords, generate a sentence using all keywords, in any order and any form of inflections. e.g.,

Input: snow drive car

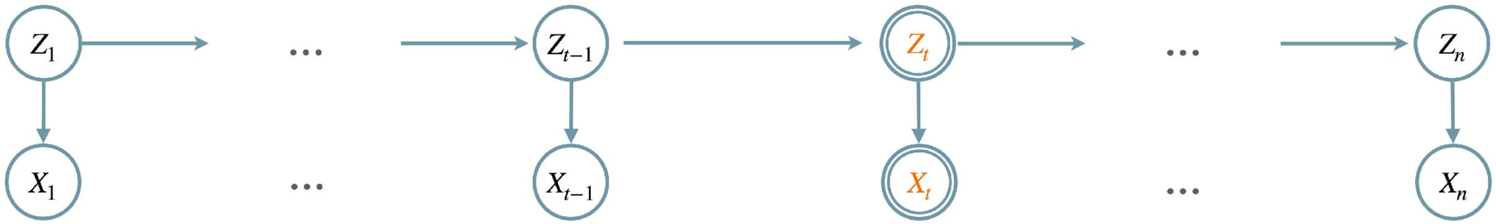
Reference 1: A car drives down a snow covered road.

Reference 2: Two cars drove through the snow.

Constraint α in CNF: $(w_{1,1} \vee \ldots \vee w_{1,d1}) \wedge \ldots \wedge (w_{m,1} \vee \ldots \vee w_{m,dm})$

Each clause represents the inflections for one keyword.

Distill an HMM p_{hmm} that approximates p_{gpt}



1. HMM with 4096 hidden states and 50k emission tokens, minimizing $KL(p_{gpt} // p_{HMM})$

2. Leverages latent variable distillation for training PCs at scale [ICLR 23].

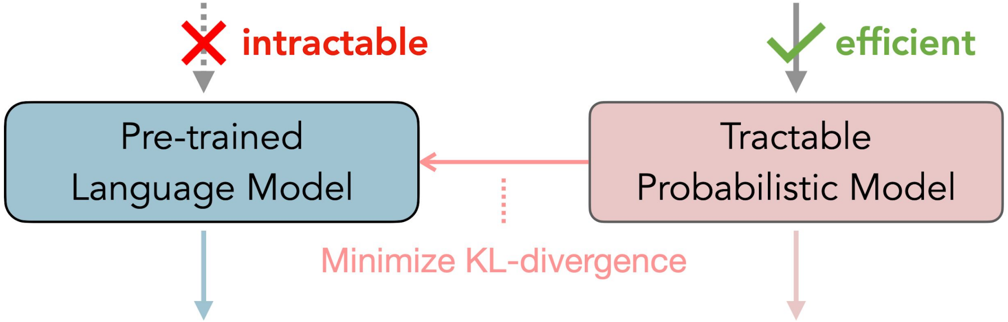
3. Efficient algorithm for computing $p(\alpha \mid x_{1:t+1})$ with constraint α in CNF:
   For m clauses and sequence length n, time-complexity for HMM generation is $O(2^{|m|}n)$

Anji Liu, Honghua Zhang and Guy Van den Broeck. Scaling Up Probabilistic Circuits by Latent Variable Distillation, 2023.

GeLaTo Overview

Lexical Constraint α: sentence contains keyword "winter"

Constrained Generation: $\Pr(x_{t+1} \mid \alpha, x_{1:t} = \text{"the weather is"})$



x_{t+1}	$\Pr_{LM}(x_{t+1} \mid x_{1:t})$
cold	0.05
warm	0.10

x_{t+1}	$\Pr_{TPM}(\alpha \mid x_{t+1}, x_{1:t})$
cold	0.50
warm	0.01

Honghua Zhang, Meihua Dang, Nanyun Peng and Guy Van den Broeck. Tractable Control for Autoregressive Language Generation, 2023.

GeLaTo Overview

Lexical Constraint α: sentence contains keyword "winter"

Constrained Generation: $\Pr(x_{t+1} | \alpha, x_{1:t} =$ "the weather is")

intractable → Pre-trained Language Model

efficient → Tractable Probabilistic Model

Minimize KL-divergence

x_{t+1}	$\Pr_{LM}(x_{t+1} \mid x_{1:t})$
cold	0.05
warm	0.10

x_{t+1}	$\Pr_{TPM}(\alpha \mid x_{t+1}, x_{1:t})$
cold	0.50
warm	0.01

x_{t+1}	$p(x_{t+1} \mid \alpha, x_{1:t})$
cold	0.025
warm	0.001

Honghua Zhang, Meihua Dang, Nanyun Peng and Guy Van den Broeck. Tractable Control for Autoregressive Language Generation, 2023.

Control p_{gpt} via p_{hmm}

Language model is fine-tuned to perform constrained generation (e.g. seq2seq)

we view $p_{HMM}(x_{t+1} \mid x_{1:t}, \alpha)$ and $p_{gpt}(x_{t+1} \mid x_{1:t})$ as classifiers trained for the same task with different biases; thus we generate from their *weighted geometric mean*:

$$p(x_{t+1} \mid x_{1:t}, \alpha) \propto p_{hmm}(x_{t+1} \mid x_{1:t}, \alpha)^{w} \cdot p_{gpt}(x_{t+1} \mid x_{1:t})^{1-w}$$

Method	*Generation Quality*								*Constraint Satisfaction*			
	ROUGE-L		BLEU-4		CIDEr		SPICE		Coverage		Success Rate	
Supervised	*dev*	*test*	*dev*	*test*	*dev*	*test*	*dev*	*test*	*dev*	*test*	*dev*	*test*
NeuroLogic (Lu et al., 2021)	-	42.8	-	26.7	-	14.7	-	30.5	-	97.7	-	93.9†
A*esque (Lu et al., 2022b)	-	43.6	-	28.2	-	15.2	-	30.8	-	97.8	-	97.9†
NADO (Meng et al., 2022)	44.4†	-	30.8	-	16.1†	-	**32.0**†	-	97.1	-	88.8†	-
GeLaTo	**46.0**	**45.6**	**34.1**	**32.9**	**16.7**	**16.8**	31.3	**31.9**	**100.0**	**100.0**	**100.0**	**100.0**

Honghua Zhang, Meihua Dang, Nanyun Peng and Guy Van den Broeck. Tractable Control for Autoregressive Language Generation, 2023.

Advantages of GeLaTo:

1. Constraint α is guaranteed to be satisfied:
   for any next-token x_{t+1} that would make α unsatisfiable, $p(x_{t+1} \mid x_{1:t}, \alpha) = 0$.

2. Training p_{hmm} does not depend on α,
   which is only imposed at inference (generation) time.

3. Can impose additional tractable constraints:
   - keywords follow a particular order
   - keywords appear at a particular position
   - keywords must not appear

Conclusion: you can control an intractable generative model using a tractable probabilistic circuit.

Probabilistic circuits seem awfully general.

Are all tractable probabilistic models probabilistic circuits?

Enter: Determinantal Point Processes (DPPs)

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

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

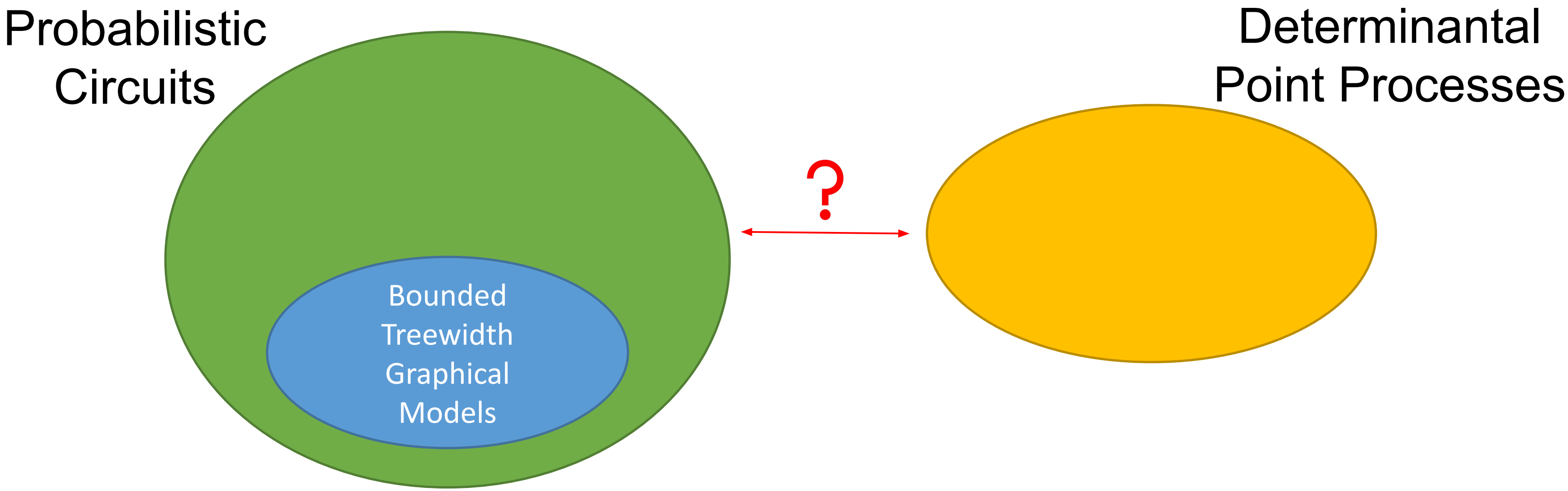
Global Negative Dependence

Diversity in recommendation systems

Tractable likelihoods and marginals

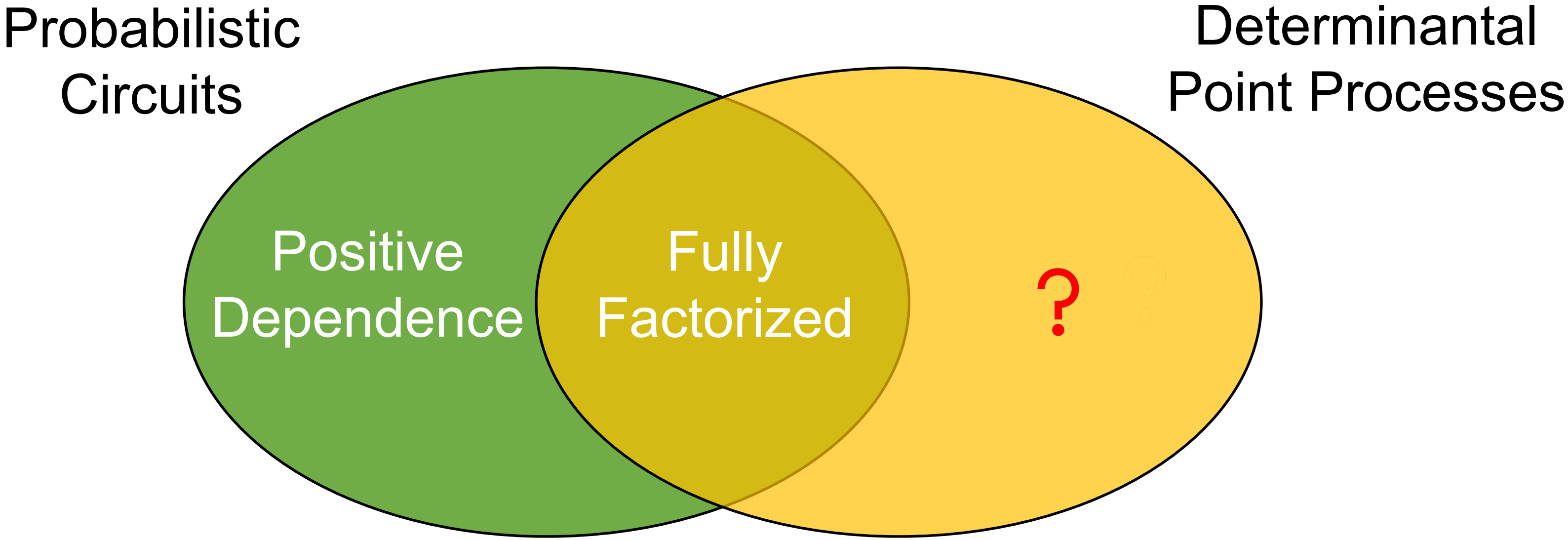
$$\Pr_L(X_1 = 1, X_2 = 0, X_3 = 1, X_4 = 0) = \frac{1}{\det(L + I)} \det(L_{\{1,2\}})$$

Are all tractable probabilistic models probabilistic circuits?



Honghua Zhang, Steven Holtzen and Guy Van den Broeck. On the Relationship Between Probabilistic Circuits and Determinantal Point Processes, *UAI*, 2020.

Are all tractable probabilistic models probabilistic circuits?

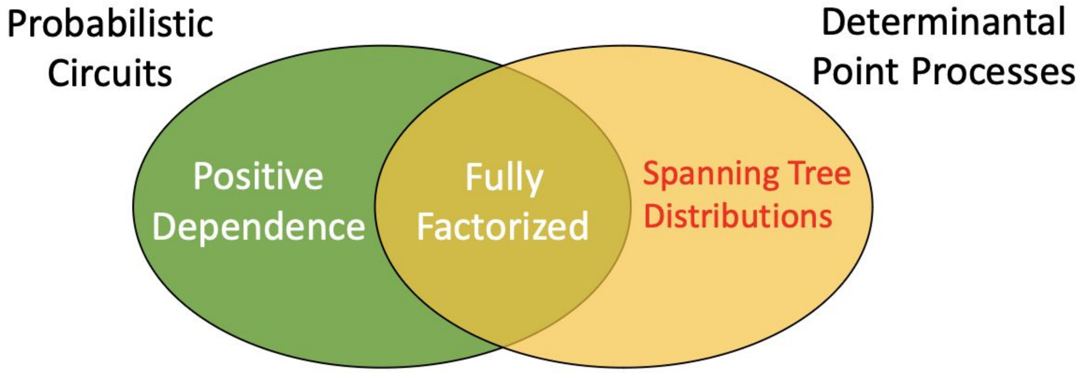


Honghua Zhang, Steven Holtzen and Guy Van den Broeck. On the Relationship Between Probabilistic Circuits and Determinantal Point Processes, *UAI*, 2020.

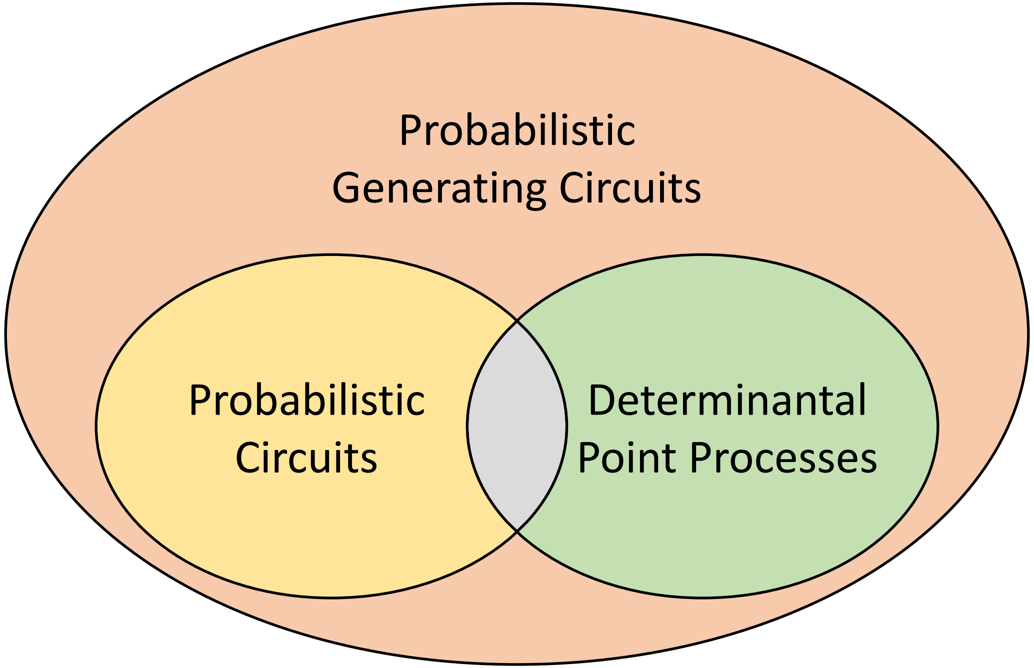
A separation between PCs and DPPs

Theorem (Martens and Medabalimi, 2014). *Let P_n be the uniform distribution over spanning trees on K_n. For $n \geq 20$, the size of any smooth and decomposable PC that represents P_n is at least $2^{n/30240}$.*

Theorem (Snell, 1995). *The uniform distribution over spanning trees on the complete graph K_n is a DPP over $\binom{n}{2}$ edges.*



Probabilistic Generating Circuits



A Tractable Unifying Framework for PCs and DPPs

Honghua Zhang, Brendan Juba and Guy Van den Broeck. Probabilistic Generating Circuits, *ICML*, 2021.

Probability Generating Functions

X_1	X_2	X_3	$\Pr_\beta$
0	0	0	0.02
0	0	1	0.08
0	1	0	0.12
0	1	1	0.48
1	0	0	0.02
1	0	1	0.08
1	1	0	0.04
1	1	1	0.16

$$g_\beta = 0.16 z_1 z_2 z_3 + 0.04 z_1 z_2 + 0.08 z_1 z_3 + 0.02 z_1 + 0.48 z_2 z_3 + 0.12 z_2 + 0.08 z_3 + 0.02.$$

Honghua Zhang, Brendan Juba and Guy Van den Broeck. Probabilistic Generating Circuits, *ICML*, 2021.

Probability Generating Functions

X_1	X_2	X_3	$\Pr_\beta$
0	0	0	0.02
0	0	1	0.08
0	1	0	0.12
0	1	1	0.48
1	0	0	0.02
1	0	1	0.08
1	1	0	0.04
1	1	1	0.16

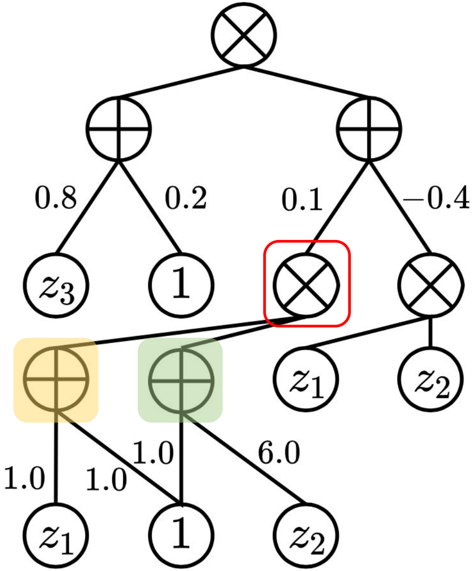
$$g_\beta = 0.16z_1z_2z_3 + 0.04z_1z_2 + 0.08z_1z_3 + 0.02z_1 + 0.48z_2z_3 + 0.12z_2 + 0.08z_3 + 0.02.$$

$$g_\beta = (0.1(z_1+1)(6z_2+1) - 0.4z_1z_2)(0.8z_3+0.2)$$

Honghua Zhang, Brendan Juba and Guy Van den Broeck. Probabilistic Generating Circuits, *ICML*, 2021.

Probabilistic Generating Circuits (PGCs)

$$g_\beta = (0.1(z_1 + 1)(6z_2 + 1) - 0.4z_1z_2)(0.8z_3 + 0.2)$$

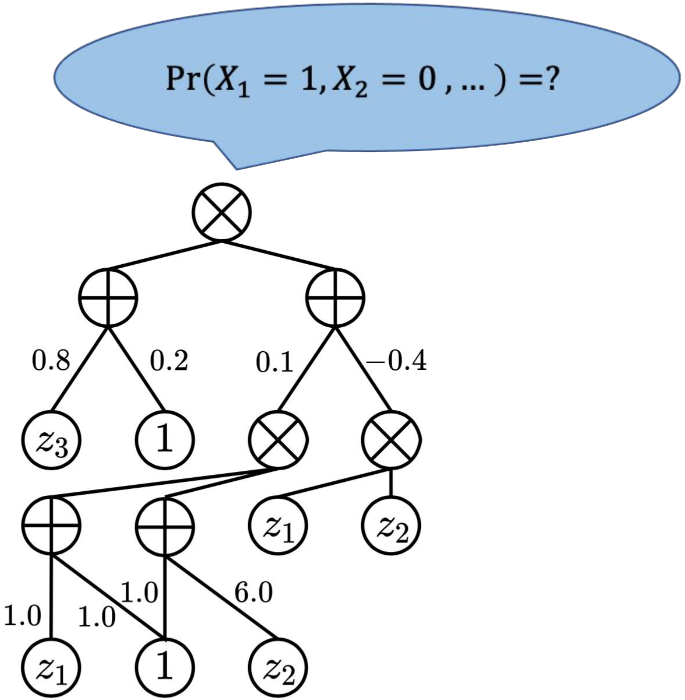


1. Sum nodes $\oplus$ with weighted edges to children.
2. Product nodes $\otimes$ with unweighted edges to children.
3. Leaf nodes: z_i or constant.

Honghua Zhang, Brendan Juba and Guy Van den Broeck. Probabilistic Generating Circuits, *ICML*, 2021.

PGCs Support Tractable **Likelihoods**

How to extract the right monomial's coefficient?



Honghua Zhang, Brendan Juba and Guy Van den Broeck. Probabilistic Generating Circuits, *ICML*, 2021.

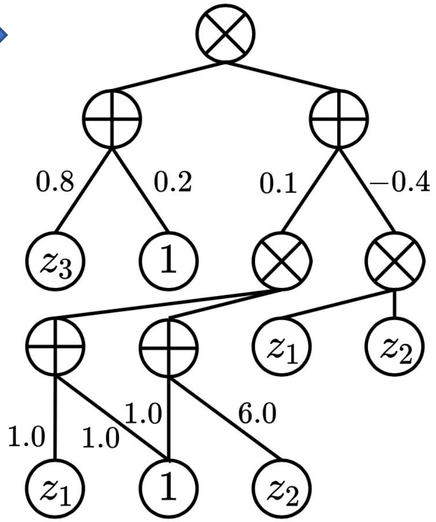
PGCs Support Tractable **Likelihoods**

How to extract the right monomial's coefficient?

Purely symbolic

$$z_i = \begin{cases} t & \text{if } X_i = 1 \\ 0 & \text{if } X_i = 0 \end{cases}$$

$\Pr(X_1 = 1, X_2 = 0, \ldots) = ?$



complexity
O(circuit size x degree)

$$p(t) = \alpha_k t^k + \cdots + \alpha_1 t$$

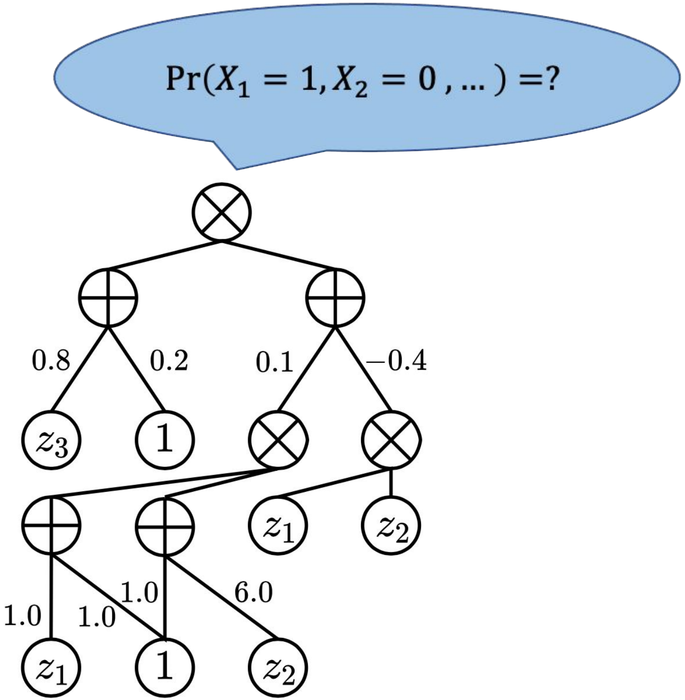
- Monomials setting to true variables that must be false are 0-ed out
- Only the monomial that sets all required variables to true has max degree.

α_k gives the answer

Honghua Zhang, Brendan Juba and Guy Van den Broeck. Probabilistic Generating Circuits, *ICML*, 2021 & Harviainen et al., UAI 2023

PGCs Support Tractable **Marginals**

How to sum the right monomial's coefficients?



Honghua Zhang, Brendan Juba and Guy Van den Broeck. Probabilistic Generating Circuits, *ICML*, 2021.

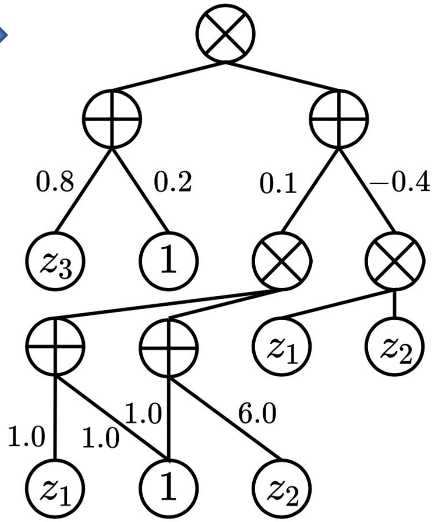
PGCs Support Tractable **Marginals**

How to sum the right monomial's coefficients?

Purely symbolic

$$z_i = \begin{cases} t & \text{if } X_i = 1 \\ 0 & \text{if } X_i = 0 \\ 1 & \text{if } X_i = ? \end{cases}$$

$\Pr(X_1 = 1, X_2 = 0, \ldots) = ?$



$$p(t) = \alpha_k t^k + \cdots + \alpha_1 t$$

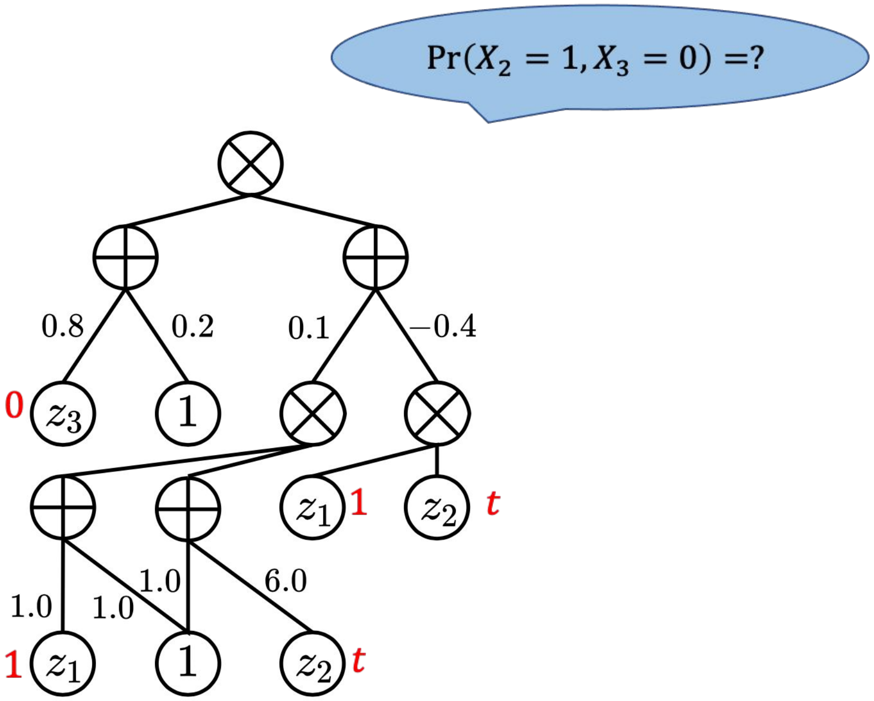
- Monomials setting to true variables that must be false are 0-ed out
- Other monomials contribute to result.
- Only monomials that set all required variables to true have max degree.

α_k gives the answer

Honghua Zhang, Brendan Juba and Guy Van den Broeck. Probabilistic Generating Circuits, *ICML*, 2021.

Example

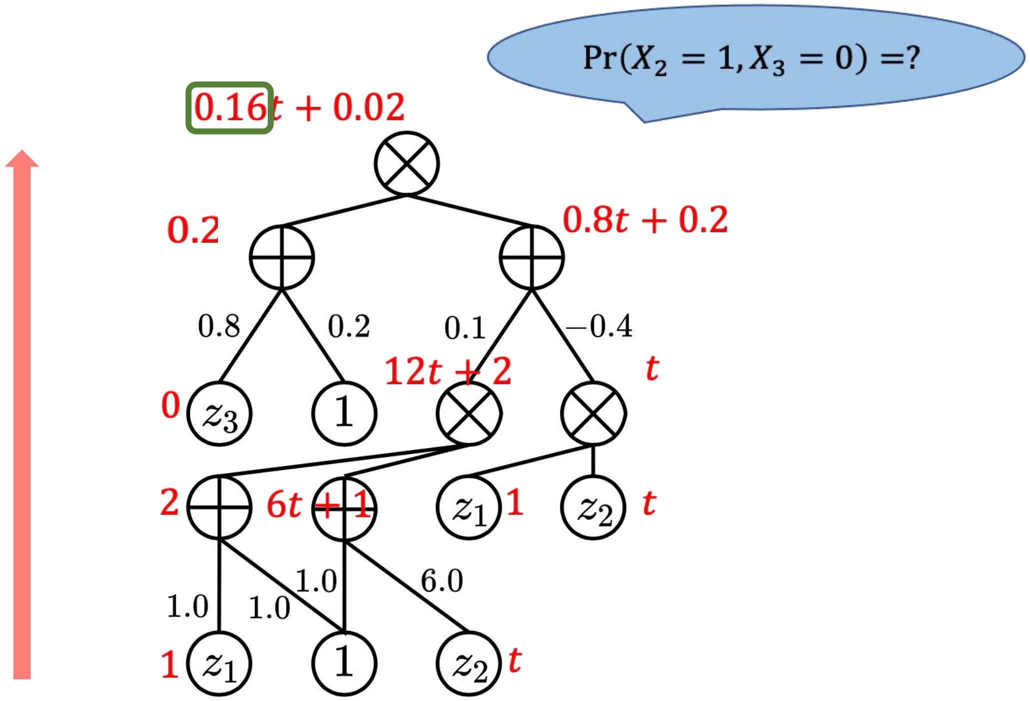
$$z_i = \begin{cases} t & \text{if } X_i = 1 \\ 0 & \text{if } X_i = 0 \\ 1 & \text{if } X_i = ? \end{cases}$$



Honghua Zhang, Brendan Juba and Guy Van den Broeck. Probabilistic Generating Circuits, *ICML*, 2021.

Example

$$z_i = \begin{cases} t & \text{if } X_i = 1 \\ 0 & \text{if } X_i = 0 \\ 1 & \text{if } X_i = ? \end{cases}$$

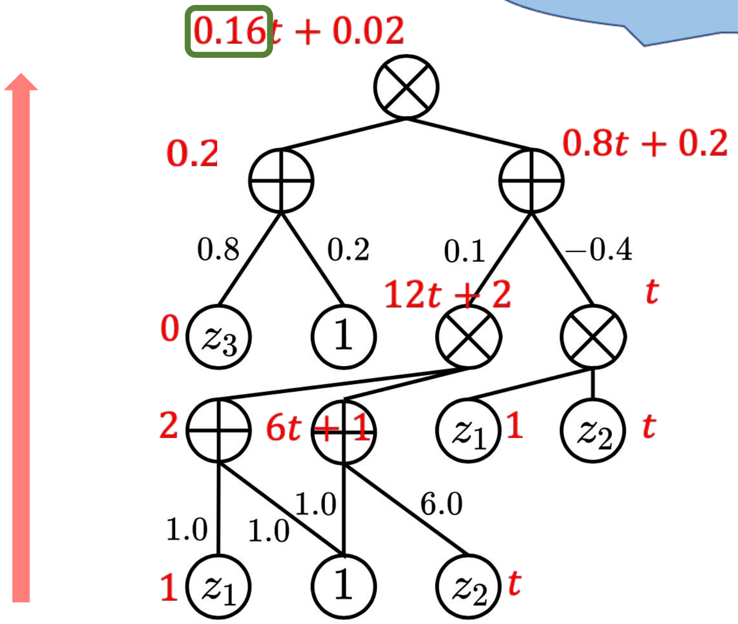


Honghua Zhang, Brendan Juba and Guy Van den Broeck. Probabilistic Generating Circuits, *ICML*, 2021.

Example

$$z_i = \begin{cases} t & \text{if } X_i = 1 \\ 0 & \text{if } X_i = 0 \\ 1 & \text{if } X_i = ? \end{cases}$$

$\Pr(X_2 = 1, X_3 = 0) = ?$



X_1	X_2	X_3	$\Pr_\beta$
0	0	0	0.02
0	0	1	0.08
0	1	0	0.12
0	1	1	0.48
1	0	0	0.02
1	0	1	0.08
1	1	0	0.04
1	1	1	0.16

Honghua Zhang, Brendan Juba and Guy Van den Broeck. Probabilistic Generating Circuits, *ICML*, 2021.

Probabilistic circuits are probabilistic generating circuits

PCs represents probability mass functions:

$$m_\beta = 0.16X_1X_2X_3 + 0.04X_1X_2\overline{X_3} + 0.08X_1\overline{X_2}X_3 + 0.02X_1\overline{X_2}\,\overline{X_3}$$
$$+ 0.48\overline{X_1}X_2X_3 + 0.12\overline{X_1}X_2\overline{X_3} + 0.08\overline{X_1}\,\overline{X_2}X_3 + 0.02\overline{X_1}\,\overline{X_2}\,\overline{X_3}$$

PGCs represent probability generating functions:

$$g_\beta = 0.16z_1z_2z_3 + 0.04z_1z_2 + 0.08z_1z_3 + 0.02z_1$$
$$+ 0.48z_2z_3 + 0.12z_2 + 0.08z_1z_3 + 0.02$$

Given a smooth & decomposable PC, by setting $\overline{X}_i$ to 1, and X_i to z_i,
we obtain an equivalent PGC

Honghua Zhang, Brendan Juba and Guy Van den Broeck. Probabilistic Generating Circuits, *ICML*, 2021.

DPPs are probabilistic generating circuits

The generating polynomial for a DPP with kernel L is given by:

$$g_L = \frac{1}{\det(L+I)} \det(I + L\mathrm{diag}(z_1, \dots, z_n)).$$

We need it as a sum of products to obtain a Probabilistic Generating Circuit

Honghua Zhang, Brendan Juba and Guy Van den Broeck. Probabilistic Generating Circuits, *ICML*, 2021.

DPPs are probabilistic generating circuits

The generating polynomial for a DPP with kernel L is given by:

$$g_L = \frac{1}{\det(L+I)} \det(I + L\mathrm{diag}(z_1, \ldots, z_n)).$$

Constant

We need it as a sum of products to obtain a Probabilistic Generating Circuit

Honghua Zhang, Brendan Juba and Guy Van den Broeck. Probabilistic Generating Circuits, *ICML*, 2021.

DPPs are probabilistic generating circuits

The generating polynomial for a DPP with kernel L is given by:

$$g_L = \frac{1}{\det(L+I)} \det(I + L\mathrm{diag}(z_1, \ldots, z_n)).$$

Constant

Division-free determinant algorithm
(Samuelson-Berkowitz algorithm)

g_L can be represented as a PGC of size $O(n^4)$

Honghua Zhang, Brendan Juba and Guy Van den Broeck. Probabilistic Generating Circuits, *ICML*, 2021.

Probabilistic **generating** circuits seem awfully general.

Are all tractable probabilistic models probabilistic ***generating*** *circuits?*

Beyond marginal probabilities

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

Antonio Vergari, YooJung Choi, Anji Liu, Stefano Teso and Guy Van den Broeck. A Compositional Atlas of Tractable Circuit Operations for Probabilistic Inference, *NeurIPS*, 2021.

Conclusions

1. What are probabilistic circuits?

   tractable deep generative models

2. What are they useful for?

   controlling generative AI

3. What is the underlying theory?

   probability generating polynomials

Thanks

This was the work of many wonderful students/postdocs/collaborators!

References: http://starai.cs.ucla.edu/publications/