PROBABILISTIC AND DIFFERENTIABLE PROGRAMMING

V10: Probabilistic Circuits I

Özgür L. Özçep

Universität zu Lübeck

Institut für Informationssysteme

Today's Agenda

Probabilistic Circuits

1. Motivation
2. Building Blocks
3. Structural Properties for Tractability

MOTIVATION

Have the cake and eat it too

- Be as expressive as possible
  - Express arbitrary distributions
- Be as efficient as possible in inferencing/answering queries
- For exact (not just approximative) answering

Tractable Prbabilistic Inference

A class of queries Q is tractable on a family of probabilistic models M iff for any query $q \in Q$ and model $m \in M$ exactly computing $q(m)$ runs in time $O(poly(|m|))$.

- Often poly will in fact be linear
- Note: if M and Q are compact in the number of random variables $\boldsymbol{X}$, i.e., $|\mathrm{m}|, |\mathrm{q}| \in \mathrm{O}(\mathrm{poly}(|\mathbf{X}|))$, then query time is $\mathrm{O}(\mathrm{poly}(|\mathrm{X}|))$.

Why exact inference?

1. No need for approximations when we can be exact
2. We can do exact inference in approximate models (e.g., Dechter et al. 2002)
3. Approximations shall come with guarantees
4. Approximate inference (even with guarantees) can mislead learners (Kulesza/Pereira 2007)
5. Approximations can be intractable as well (Dagum/Luby1993)

Complete Evidence (EVI)

- q_3: *What is the probability that today is a Monday at 12.00 and there is a traffic jam only on 5th Avenue?*

- $X = \{Day, Time, Jam_{5th}, Jam_{Str2}, \ldots, Jam_{StrN}\}$

- $q_3(m) = p_m(X = \{Mon, 12.00, 1, 0, \ldots, 0\})$

- Fundamental in maximum likelihood learning

$$\theta_m^{MLE} = argmax_\theta(\Pi_{x \in D} p_m(x; \theta))$$

© fineartamerica.com

Marginal (MAR) and Conditional (CON) queries

- q_1: *What is the probability that today is a Monday ~~at 12.00~~ and there is a traffic jam ~~only~~ on 5th Avenue?*

- $q_1(m) = p_m(Day = Mon, Jam_{5th} = 1\ )$

- General:

$$p_m(e) = \int p_m(e, \boldsymbol{H}) d\boldsymbol{H}$$

- With this can answer conditional queries too

$$p_m(q \mid \boldsymbol{e}) = \frac{p_m(q, \boldsymbol{e})}{p_m(\boldsymbol{e})}$$

© fineartamerica.com

Maximum A Posteriori (MAP) (aka Most Probable Explanation (MPE))

- q_5: *Which combination of roads is most likely to be jammed on Monday at 9am*?
- $q_5(m) = argmax_{\boldsymbol{j}} p_m(j_1, j_2, \ldots \mid Day = Mon, Time = 9)$
- General: $argmax_{\boldsymbol{q}} \; p_m(\boldsymbol{q} \mid \boldsymbol{e})$
  where $\boldsymbol{Q} \cup \boldsymbol{E} = \boldsymbol{X}$

© fineartamerica.com

Maximum A Posteriori (MAP) (aka Most Probable Explanation (MPE))

- q_5: Which combination of roads is most likely to be jammed on Monday at 9am?

- ... Intractable for latent variable models

$$\max_{q} p_m(\boldsymbol{q} \mid \boldsymbol{e}) = \max_{\boldsymbol{q}} \sum_{\boldsymbol{z}} p_m(\boldsymbol{q}, \boldsymbol{z} \mid \boldsymbol{e})$$

$$\neq \sum_{\boldsymbol{z}} \max_{\boldsymbol{q}} p_m(\boldsymbol{q}, \boldsymbol{z} \mid \boldsymbol{e})$$

© fineartamerica.com

Marginal MAP (MMAP) (aka Bayesian network MAP)

- q_6: Which combination of roads is most likely to be jammed ~~on Monday~~ at 9am?
- $q_6(m) =$
  $$argmax_{\boldsymbol{j}}\, p_m(j_1, j_2, \ldots \mid, Time = 9)$$
- General: $argmax_{\boldsymbol{q}}\ p_m(\boldsymbol{q} \mid \boldsymbol{e})$
  $$= armax_{\boldsymbol{q}}\, \Sigma_h p_m(\boldsymbol{q}, \boldsymbol{h} \mid \boldsymbol{e})$$
  where $\boldsymbol{Q} \cup \boldsymbol{H} \cup \boldsymbol{E} = \boldsymbol{X}$

© fineartamerica.com

- NP^{PP}-complete (Park/Darwiche)
- NP-hard for trees (Campos 2011)

Advanced Queries (ADV)

- q_2: Which day is most likely to have a traffic jam on my route to work?
- $q_2(m) = argmax_d\ p_m(Day = d, \wedge \vee_{i \in route}\ Jam_{Stri})$
  - => Marginals + MAP + logical events
- q_7: What is the probability of seeing more traffic jams in Uptown than Midtown?
  - => counts + group comparison

© fineartamerica.com

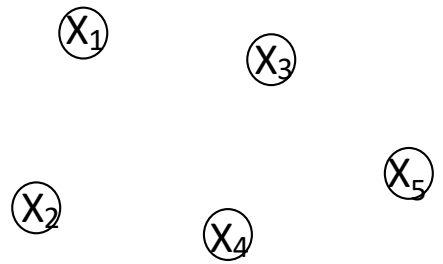
- And more
  - expected classification agreement (Oztok et al. 2016)
  - Expected predictions (Khosravi et al. 2019b)

tractable bands

1) (Kobyzev et al. 19)

OK, fully factorized models have broadest tractability spectrum, but ...

A completely disconnected graph. Example: Product of Bernoullis (PoBs)



$$p(x) = \prod_{i=1}^{n} p(x_i)$$

Complete evidence, marginals and MAP, MMAP inference is **linear**!

⇒ but definitely not expressive…

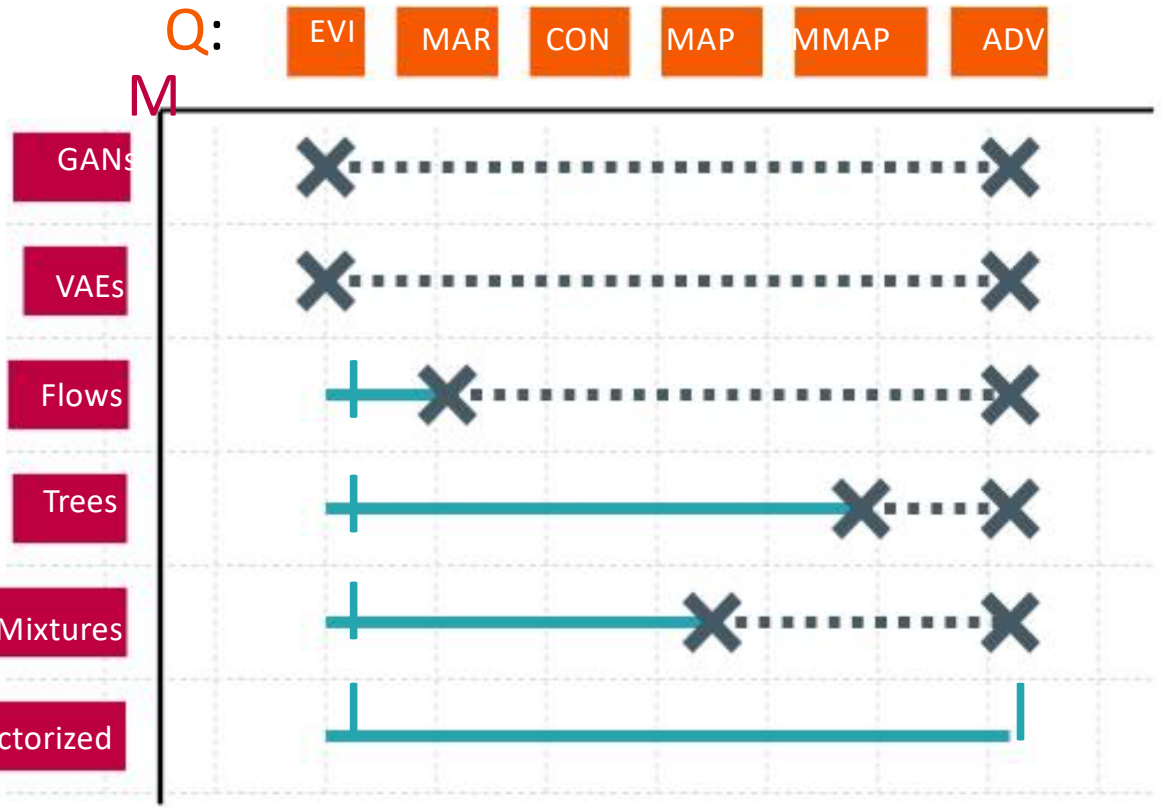
(no dependencies represetable)

Expressiveness and efficiency

- Expressiveness: Ability to represent rich and effective classes of functions

- Mixture of Gaussians can approximate any distribution!
  - See (Cohen et al. 15)
- Expressive efficiency (succinctness) Ability to represent rich and effective classes of functions compactly

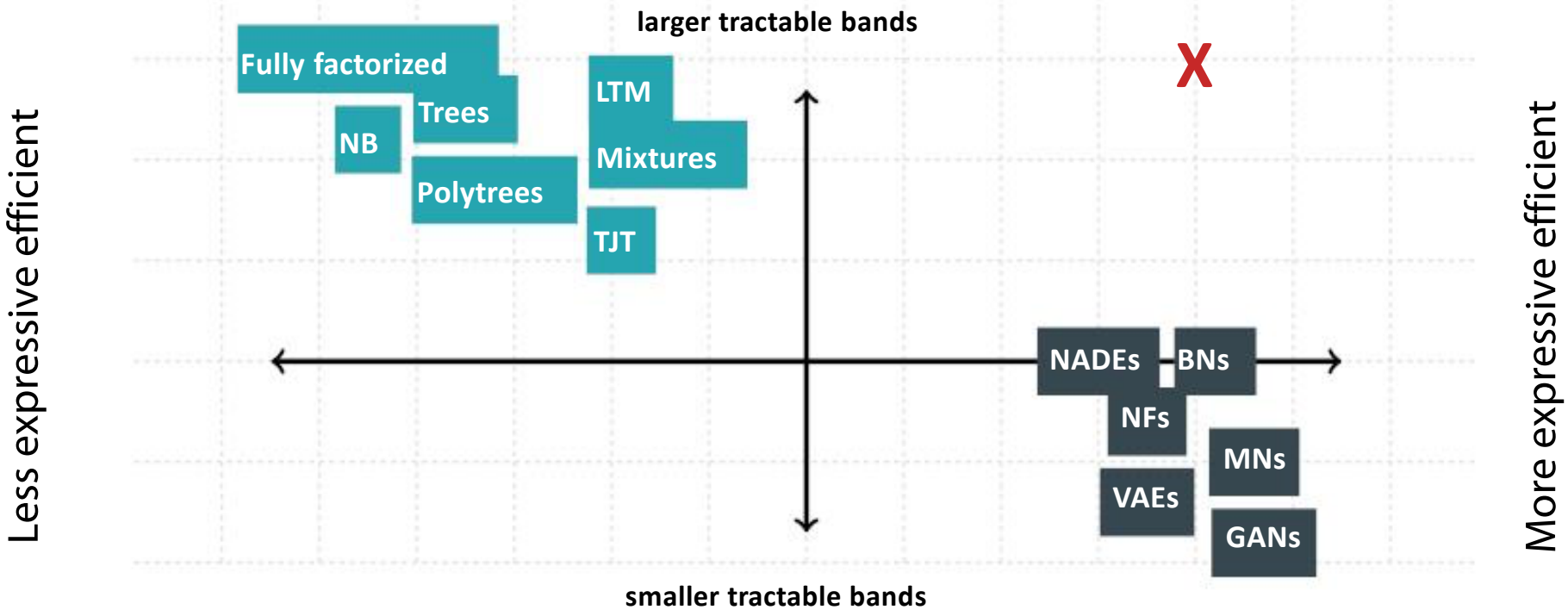
- ⇒ but how many components does a Gaussian mixture need*?*



Use these as building blocks

„Eat the cake and have it“

tractable bands



probabilistic circuits are at the "sweet spot"

BUILDING BLOCKS

A probabilistic circuit C over variables $\boldsymbol{X}$ is a computational graph encoding a (possibly unnormalized) probability distribution $p(\boldsymbol{X})$.

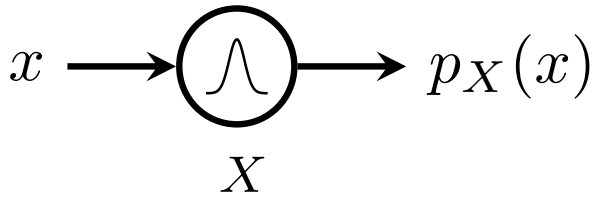
- Note that we have an operational semantics here
- By constraining the graph one can make inference tractable

Distributions as computational graphs

Base case: a single node encoding a distribution

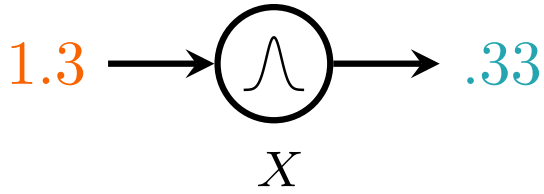
e.g., a Gaussian PDF continuous variable

e.g., indicators for X or $\neg X$ for Boolean RVs



Simple distributions are tractable „black boxes“ for

- EVI: output p(x) (density or mass)
- MAR: output 1 (normalized) or Z (unnormalized)
- MAP: output the mode



Simple distributions are tractable „black boxes“ for

- EVI: output p(x) (density or mass) <=
- MAR: output 1 (normalized) or Z (unnormalized)
- MAP: output the mode

Reminder: Partition function Z

$$P(X_1, \dots X_n) = \frac{1}{Z} \prod_j \phi_j (X_1, \dots X_n)$$

- Bottleneck: Summing out variables
- E.g.: Partition function

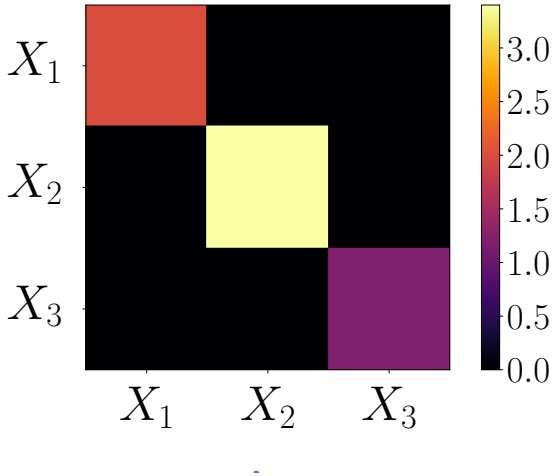
  Sum of exponentially many products

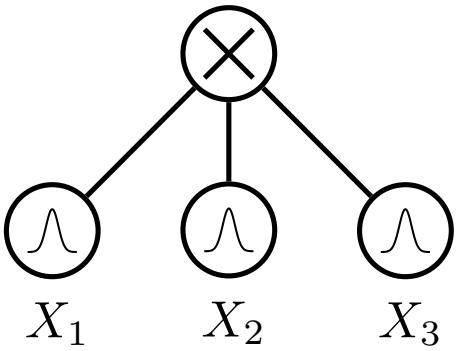
$$Z = \sum_x \prod_j \phi_j$$

Factorizations as product nodes

(Divide and Conquer complexity)

$$p(X_1, X_2, X_3) = p(X_1) \cdot p(X_2) \cdot p(X_3)$$



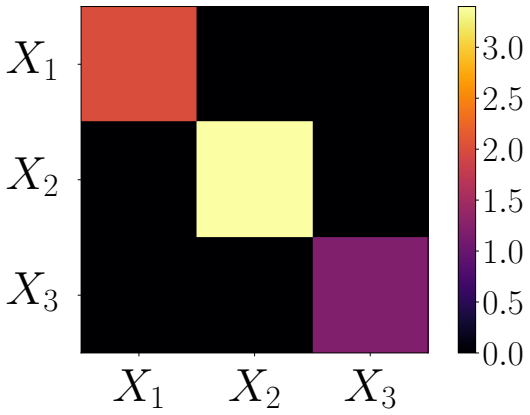


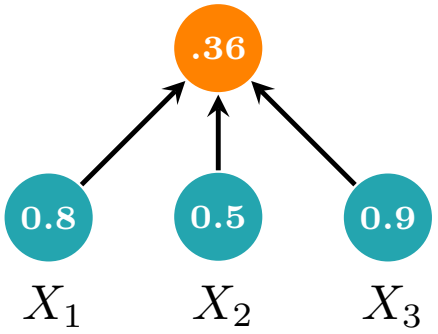
e.g. modeling a multivariate Gaussian with diagonal covariance matrix by a product node of univariate Gaussians

Factorizations as product nodes

(Divide and Conquer complexity)

$$p(x_1, x_2, x_3) = p(x_1) \cdot p(x_2) \cdot p(x_3)$$

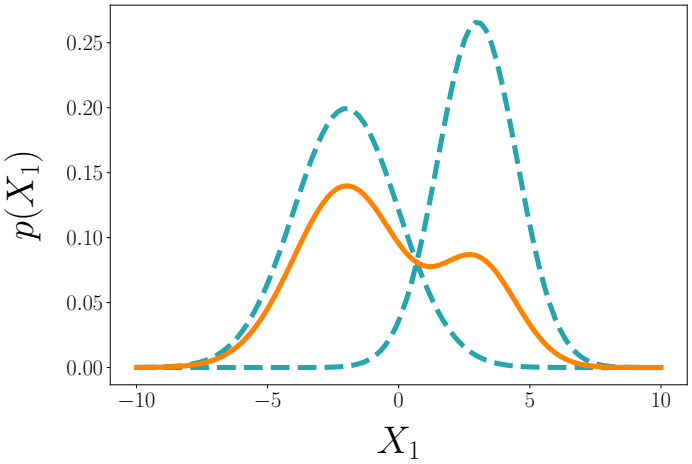




Feed forward evaluation

Mixtures as sum nodes

(enhance expressiveness)

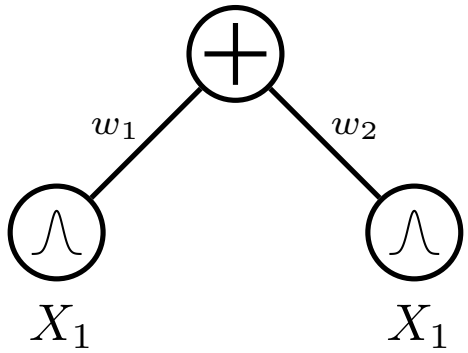


$$p(X) = w_1 \cdot p_1(X) + w_2 \cdot p_2(X)$$

⇒ *e.g. modeling a mixture of Gaussians…*

Mixtures as sum nodes

(enhance expressiveness)

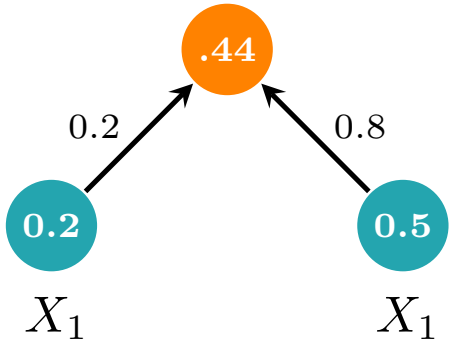


$$p(x) = 0.2 \cdot p_1(x) + 0.8 \cdot p_2(x)$$

$\Rightarrow$ *...as weighted sum node over Gaussian input distributions*

Mixtures as sum nodes

(enhance expressiveness)

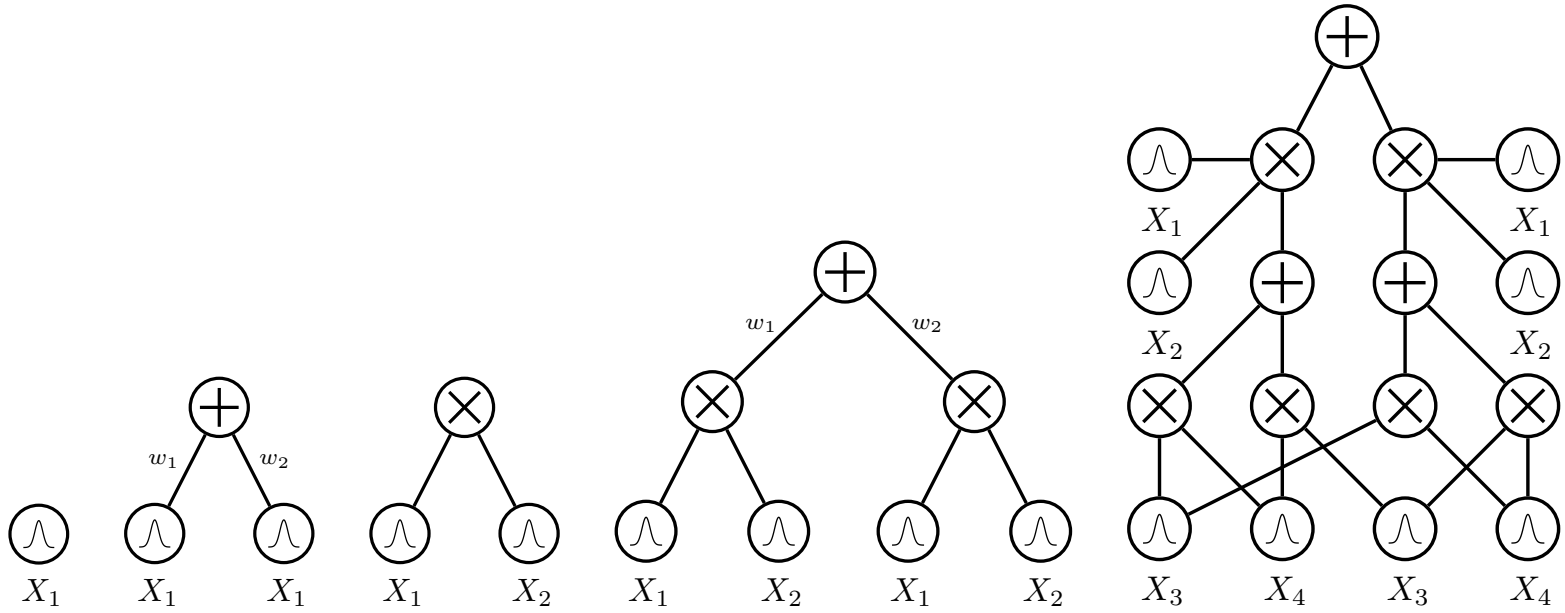


$$p(x) = 0.2 \cdot p_1(x) + 0.8 \cdot p_2(x)$$

$\Rightarrow$ *by **stacking** them we increase expressive efficiency*

A grammar for tractable models

(Recursive Semantics for probabilistic circuits)



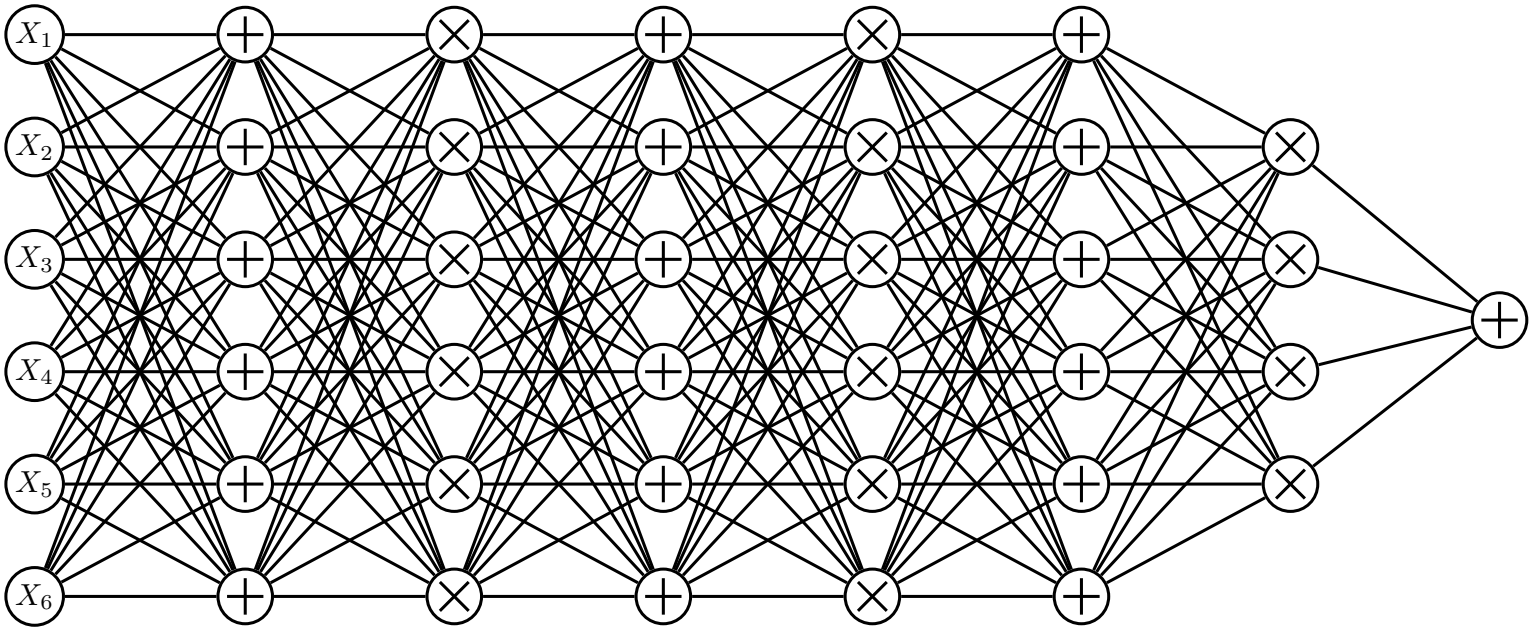
Probabilistic Circuits are not PGMs

They are ***probabilistic*** and ***graphical***, however …

	PGMs	***Circuits***
Nodes:	random variables	unit of computations
Edges:	dependencies	order of execution
Inference:	■ conditioning ■ elimination ■ message passing	■ feedforward pass ■ backward pass

⇒ *they are **computational graphs**, more like neural networks*

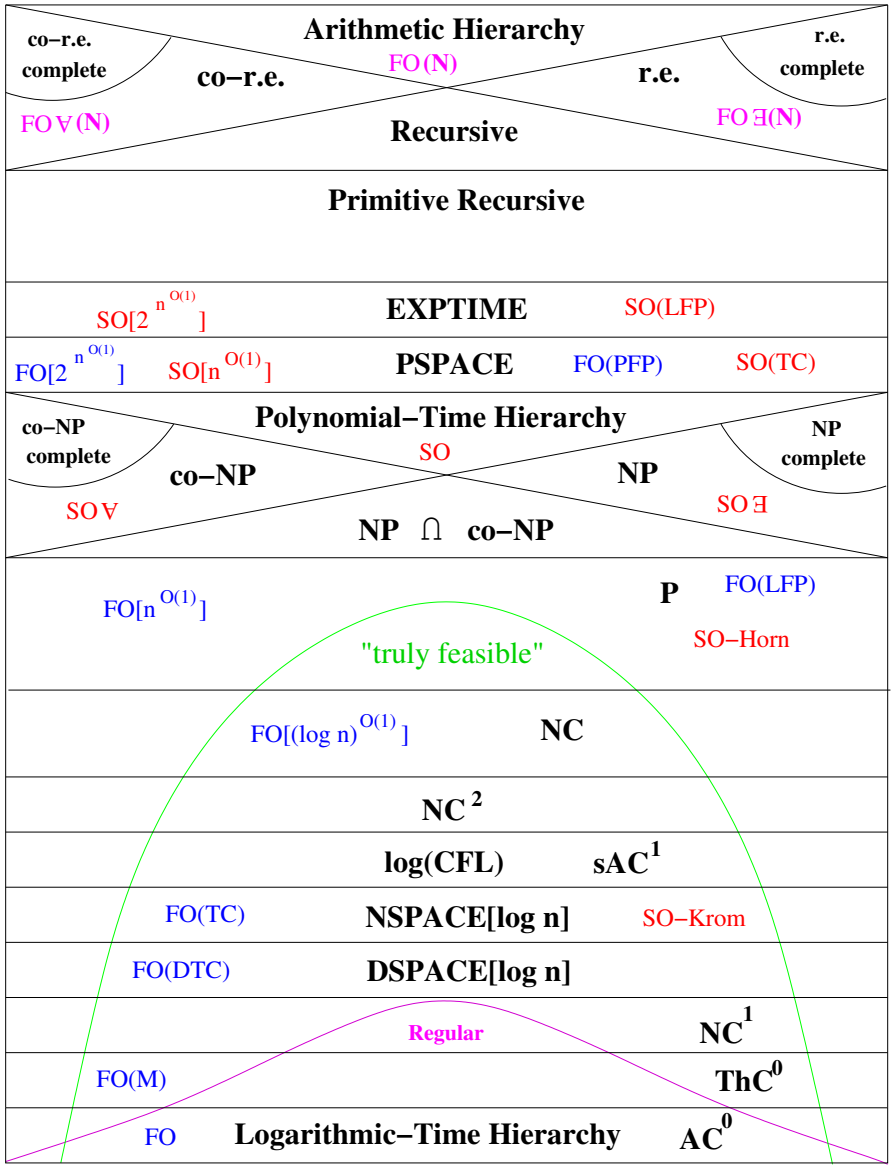
Control on the graph



- We do not arbitraly compose the building bl ocks as in neural networks
- But define structural constraints for tractability

Side note: Compare this with desriptive complexity

Correspondence of
logics and complexity classes

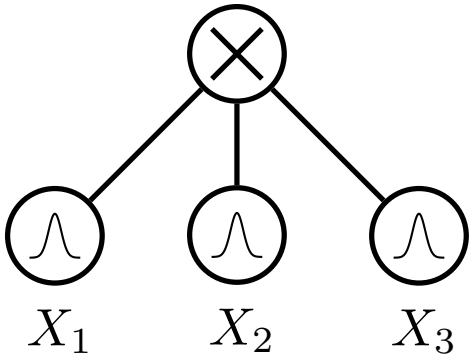


The Descriptive World

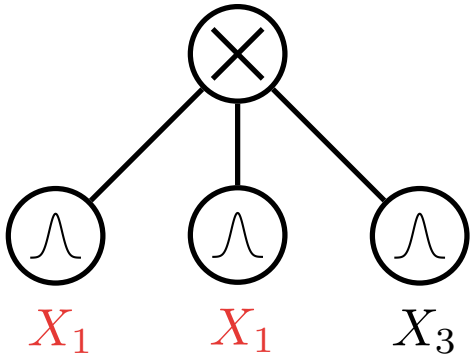
STRUCTURAL PROPERTIES FOR TRACTABILITY

Decomposability

A product node is decomposable if its children depend on disjoint sets of variables (*just like in factorization*)



decomposable circuit

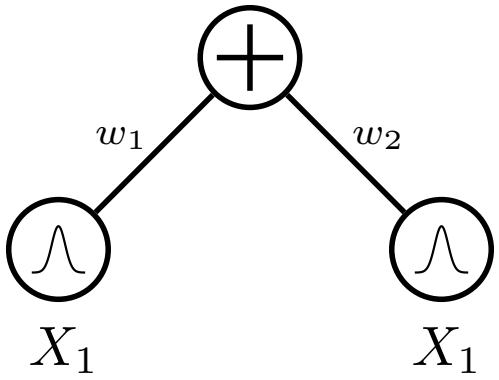


***non*-decomposable** circuit

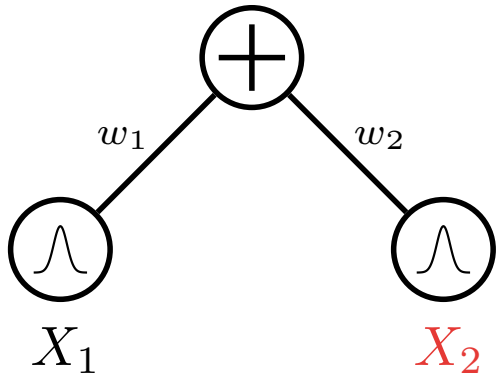
(Darwiche/Marquis 01)

Smoothness (aka as completeness)

A sum node is smooth iff its children depend on the same variable sets (*otherwise not accounting for some variables*)



smooth circuit



non-smooth circuit

(Darwiche/Marquis 01)

Smoothness + decomposability = tractable MAR

Computing arbitrary integrations (or summations)

$\Rightarrow$ *linear in circuit size!*

E.g., suppose we want to compute Z:

$$\int \boldsymbol{p}(\mathbf{x}) d\mathbf{x}$$

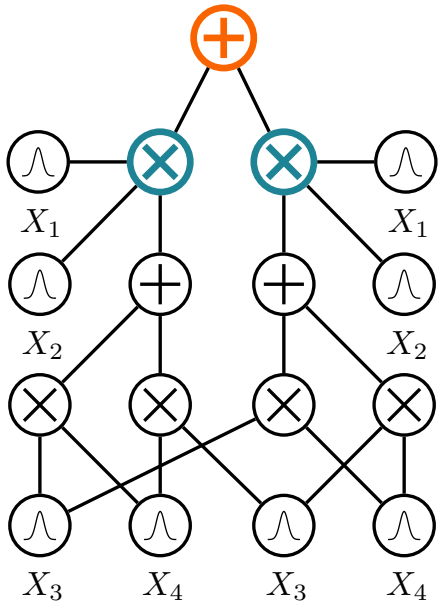
Smoothness + decomposability = tractable MAR

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

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

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

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

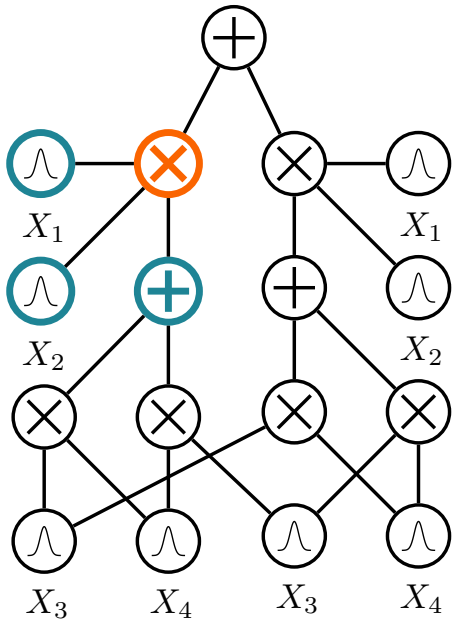


Smoothness + decomposability = tractable MAR

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

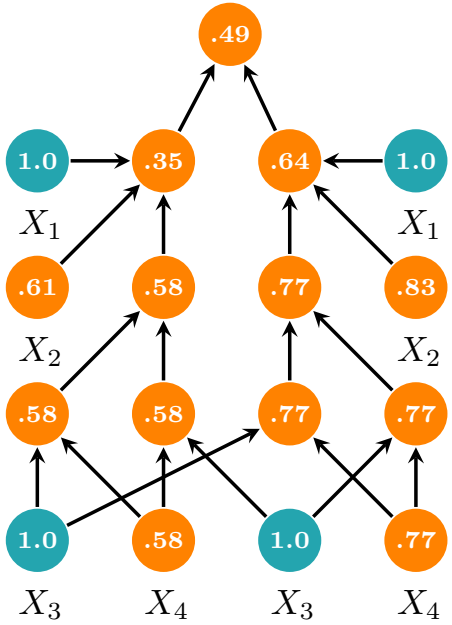
$\Rightarrow$ *linear in circuit size!*

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

- leaves over X_1 and X_3 output $Z_i = \int p(x_i) dx_i$

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

- leaves over X_2 and X_4 output **EVI**
- feedforward evaluation (bottom-up)



Analogously one can show:
Smoothness + decomposability = tractable CON

Note: Nodes with the same RV-label may have different probabilities associated with them. Hence, e.g., the left bottom X_4 may get a different value than the right bottom X_4

Smoothness + decomposability ≠ tractable MAP

We ***cannot*** decompose bottom-up a MAP query:

$$\operatorname*{argmax}_{\mathbf{q}} p(\mathbf{q} \mid \mathbf{e})$$

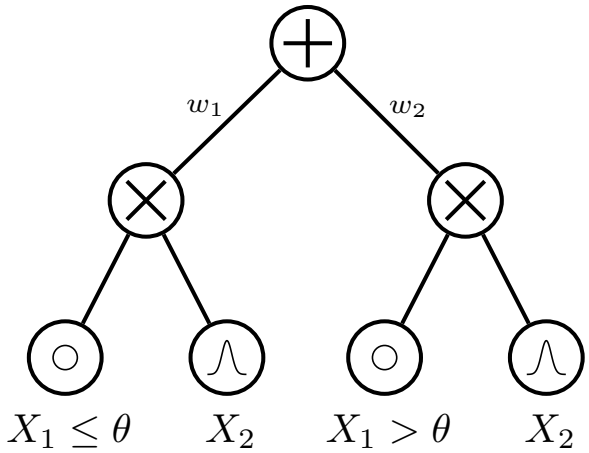
since for a sum node we are marginalizing out a latent variable

$$\operatorname*{argmax}_{\mathbf{q}} \sum_i w_i p_i(\mathbf{q}, \mathbf{e}) = \operatorname*{argmax}_{\mathbf{q}} \sum_{\mathbf{z}} p(\mathbf{q}, \mathbf{z}, \mathbf{e}) \neq \sum_{\mathbf{z}} \operatorname*{argmax}_{\mathbf{q}} p(\mathbf{q}, \mathbf{z}, \mathbf{e})$$

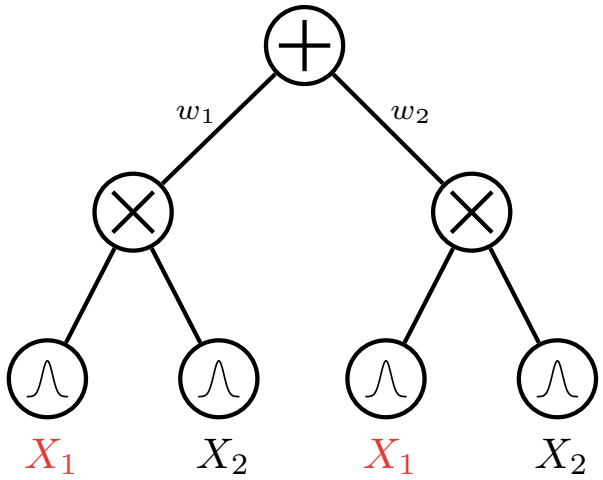
$\Rightarrow$ *MAP for latent variable models is* **intractable** *[Conaty et al. 2017]*

Determinism (aka selectivity)

A sum node is deterministic if the output of only one of its children is non zero for any input *(e.g. if their distributions have disjoint support)*



deterministic circuit



***non*-deterministic circuit**

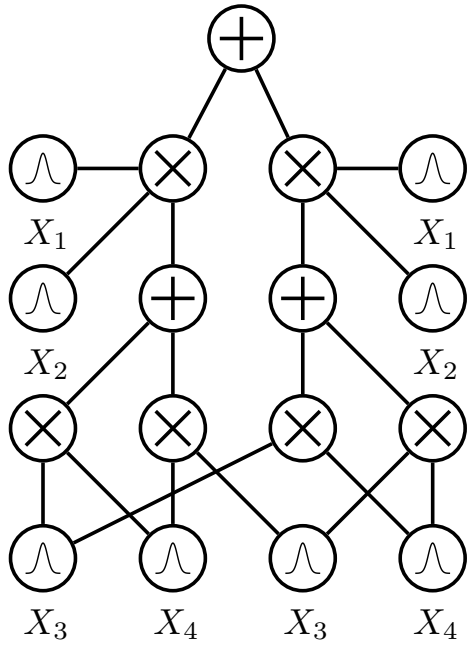
Determinism + decomposability = tractable MAP

Computing maximization with arbitrary evidence $\mathbf{e}$

$\Rightarrow$ *linear in circuit size!*

E.g., suppose we want to compute:

$$\max_{\mathbf{q}} p(\mathbf{q} \mid \mathbf{e})$$

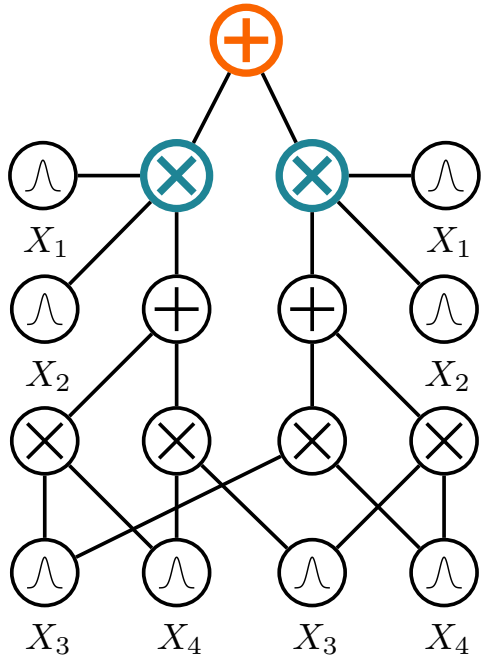


Determinism + decomposability = tractable MAP

If $p(\mathbf{q}, \mathbf{e}) = \sum_i w_i p_i(\mathbf{q}, \mathbf{e}) = \max_i w_i p_i(\mathbf{q}, \mathbf{e})$,
(***deterministic*** sum node):

$$\begin{aligned}\max_{\mathbf{q}} p(\mathbf{q}, \mathbf{e}) &= \max_{\mathbf{q}} \sum_i w_i p_i(\mathbf{q}, \mathbf{e}) \\ &= \max_{\mathbf{q}} \max_i w_i p_i(\mathbf{q}, \mathbf{e}) \\ &= \max_i \max_{\mathbf{q}} w_i p_i(\mathbf{q}, \mathbf{e})\end{aligned}$$

$\Longrightarrow$ *one non-zero child term, thus sum is max*

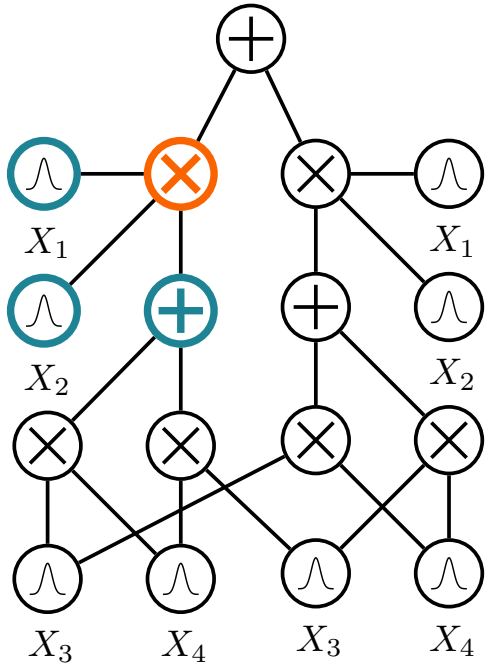


Determinism + decomposability = tractable MAP

If $p(\mathbf{q}, \mathbf{e}) = p(\mathbf{q_x}, \mathbf{e_x}, \mathbf{q_y}, \mathbf{e_y}) = p(\mathbf{q_x}, \mathbf{e_x})p(\mathbf{q_y}, \mathbf{e_y})$
(***decomposable*** product node):

$$\begin{aligned}
\max_{\mathbf{q}} p(\mathbf{q} \mid \mathbf{e}) &= \max_{\mathbf{q}} p(\mathbf{q}, \mathbf{e}) && \cdot 1/p(\boldsymbol{e}) \\
&= \max_{\mathbf{q_x}, \mathbf{q_y}} p(\mathbf{q_x}, \mathbf{e_x}, \mathbf{q_y}, \mathbf{e_y}) && \cdot 1/p(\boldsymbol{e}) \\
&= \max_{\mathbf{q_x}} p(\mathbf{q_x}, \mathbf{e_x}), \max_{\mathbf{q_y}} p(\mathbf{q_y}, \mathbf{e_y}) && \cdot 1/p(\boldsymbol{e})
\end{aligned}$$

$\Rightarrow$ *solving optimization independently*



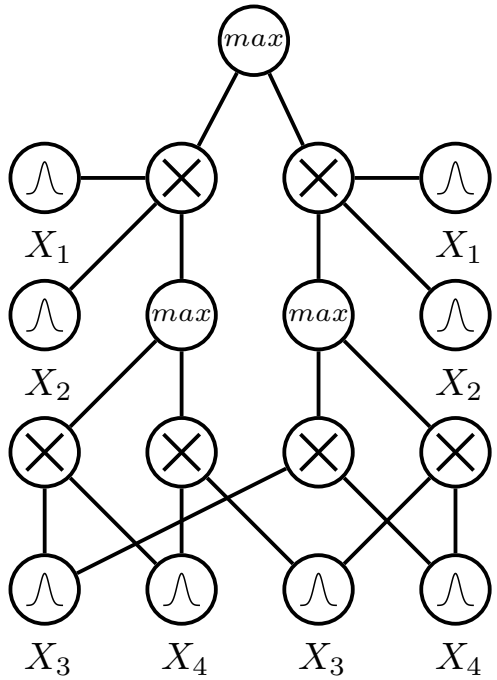
Determinism + decomposability = tractable MAP

Evaluating the circuit twice:

bottom-up and ***top-down*** $\Rightarrow$ *still linear in circuit size!*

E.g., for $\operatorname{argmax}_{x_1,x_3} p(x_1, x_3 \mid x_2, x_4)$:

1. turn sum into max nodes and distributions into max distributions
2. evaluate $p(x_2, x_4)$ bottom-up
3. retrieve max activations top-down
4. compute **MAP states** for X_1 and X_3 at leaves

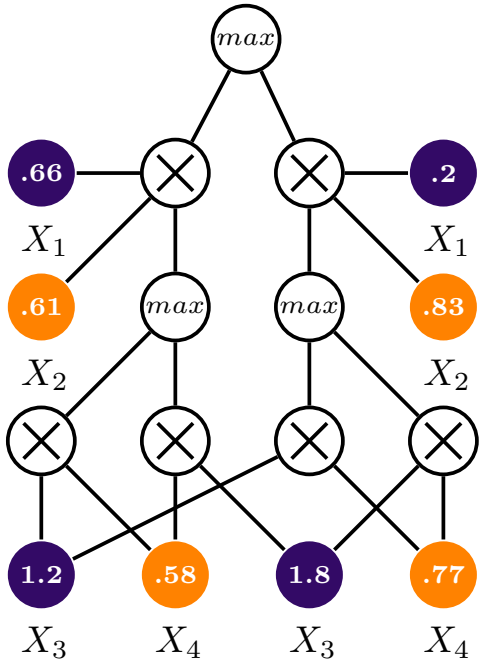


Determinism + decomposability = tractable MAP

Evaluating the circuit twice:
bottom-up and ***top-down*** $\Rightarrow$ *still linear in circuit size!*

E.g., for $\operatorname{argmax}_{x_1,x_3} p(x_1, x_3 \mid x_2, x_4)$:

1. turn sum into max nodes and distributions into max distributions
2. evaluate $p(x_2, x_4)$ bottom-up
3. retrieve max activations top-down
4. compute **MAP states** for X_1 and X_3 at leaves

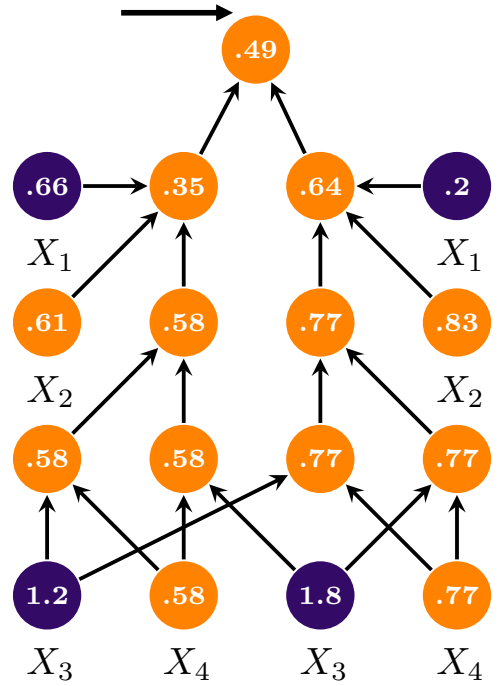


Determinism + decomposability = tractable MAP

Evaluating the circuit twice:
bottom-up and ***top-down*** $\Rightarrow$ *still linear in circuit size!*

E.g., for $\operatorname{argmax}_{x_1,x_3} p(x_1, x_3 \mid x_2, x_4)$:

1. turn sum into max nodes and distributions into max distributions
2. evaluate $p(x_2, x_4)$ bottom-up
3. retrieve max activations top-down
4. compute **MAP states** for X_1 and X_3 at leaves



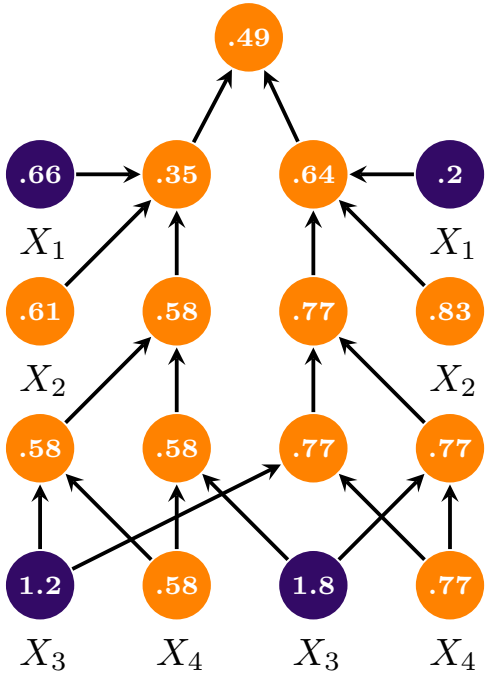
Determinism + decomposability = tractable MAP

Evaluating the circuit twice:

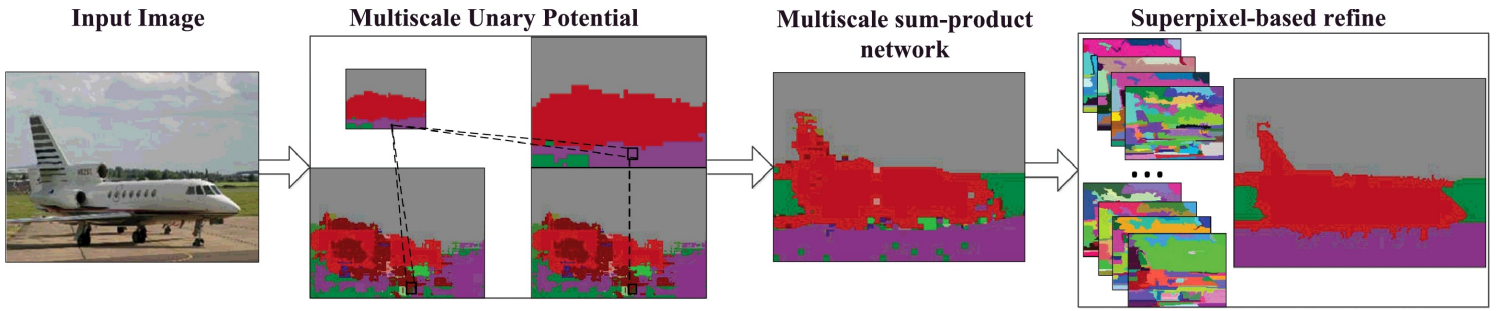
bottom-up and ***top-down*** $\Rightarrow$ *still linear in circuit size!*

E.g., for $\operatorname{argmax}_{x_1,x_3} p(x_1, x_3 \mid x_2, x_4)$:

1. turn sum into max nodes and distributions into max distributions
2. evaluate $p(x_2, x_4)$ bottom-up
3. retrieve max activations top-down
4. compute **MAP states** for X_1 and X_3 at leaves



MAP inference: image segmentation



Semantic segmentation is MAP over joint pixel and label space

Even approximate MAP for non-deterministic circuits (SPNs) delivers good performances.

Rathke et al., "Locally adaptive probabilistic models for global segmentation of pathological oct scans", 2017

Yuan et al., "Modeling spatial layout for scene image understanding via a novel multiscale sum-product network", 2016

Friesen et al., "Submodular Sum-product Networks for Scene Understanding", 2016

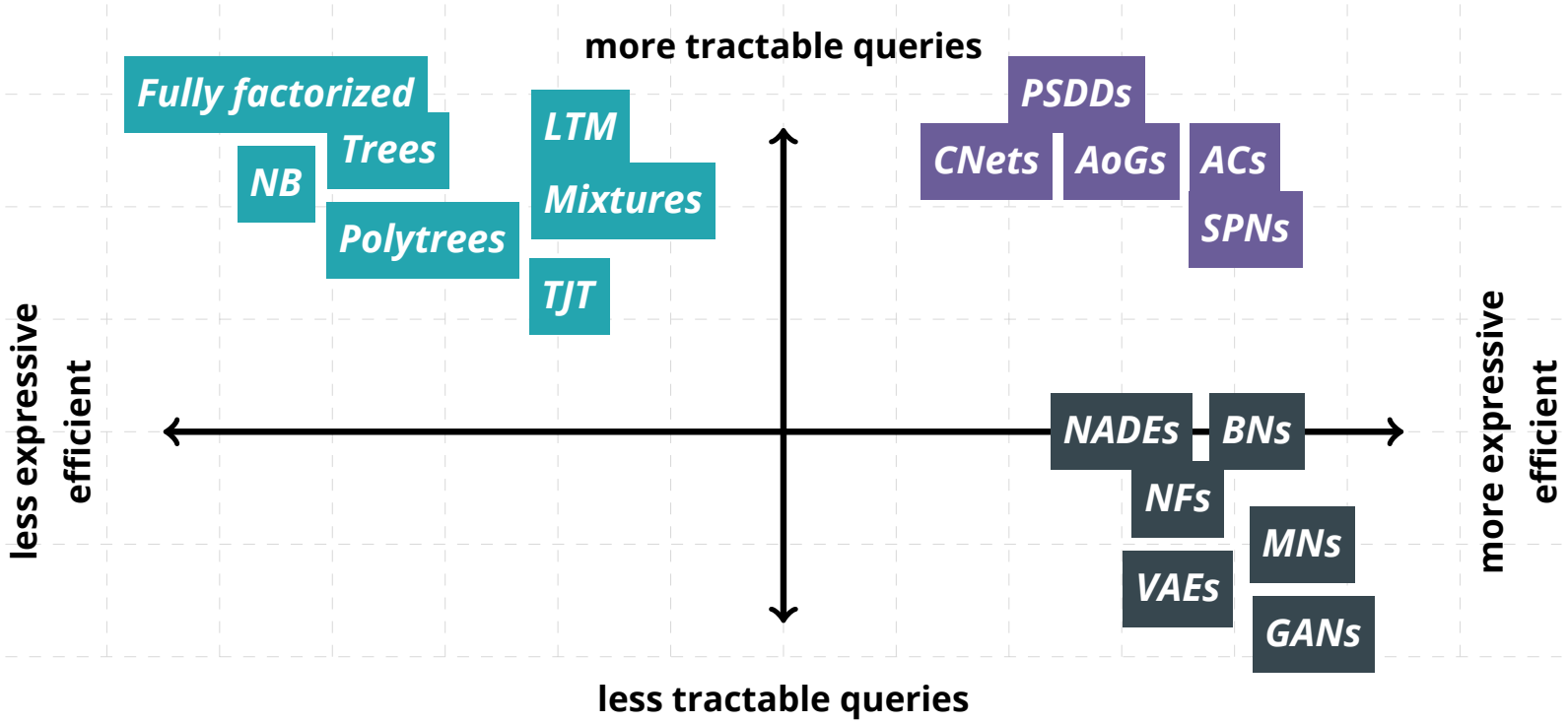
Determinism + decomposability ≠ tractable MMAP

We ***cannot*** decompose a MMAP query!

$$\operatorname*{argmax}_{\mathbf{q}} \sum_{\mathbf{z}} p(\mathbf{q}, \mathbf{z} \mid \mathbf{e})$$

we still have latent variables to marginalize...

This will be discussed in lecture V12 (when considering advanced queries)



tractability vs expressive efficiency

How expressive are probabilistic circuits?

Measuring average test set log-likelihood on 20 density estimation benchmarks

Comparing against intractable models:

- Bayesian networks (BN) *[Chickering 2002]* with sophisticated context-specific CPDs
- MADEs *[Germain et al. 2015]*
- VAEs *[Kingma et al. 2014]* (IWAE ELBO *[Burda et al. 2015]*)

Gens et al., "Learning the Structure of Sum-Product Networks", 2013

Peharz et al., "Random sum-product networks: A simple but effective approach to probabilistic deep learning", 2019

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

(Best negative log-likelihoods in bold)

Uhhh, a lecture with a hopefully useful

APPENDIX

Probability theory basics reminder

Random variable (RV)

- possible worlds defined by assignment of values to random variables.
- Boolean random variables
  e.g., Cavity (do I have a cavity?).
  Domain is < true , false >
- Discrete random variables
  e.g., possible value of Weather is one of
  < sunny, rainy, cloudy, snow >
- Domain values must be exhaustive and mutually exclusive
- Elementary propositions are constructed by assignment of a value to a random variable: e.g.,
  - Cavity = false (abbreviated as ¬cavity)
  - Cavity = true (abbreviated as cavity)
- (Complex) propositions formed from elementary propositions and standard logical connectives, e.g., Weather = sunny ∨ Cavity = false

Probabilities

- Axioms (for propositions $a, b, \top = (a \vee \neg a)$, and $\bot = \neg \top$):
  - $0 \leq P(a) \leq 1;\ P(\top) = 1;\ P(\bot) = 0$
  - $(P(a \vee b) = P(a) + P(b) - P(a \wedge b)$
- Joint probability distribution of $\mathbf{X} = \{X_1, \ldots, X_n\}$
  - $\boldsymbol{P}(X_1, \ldots, X_n)$
  - gives the probability of every atomic event on $\boldsymbol{X}$
- Conditional probability
  $P(a \mid b) = P(a \wedge b) / P(b)\ if\ P(b) > 0$
- Chain rule
  $$\boldsymbol{P}(X_1, \ldots, X_n) = \prod_{i=1}^{n} \boldsymbol{P}(X_i | X_1, \ldots, X_{i-1})$$
- Marginalization: $\boldsymbol{P}(Y) = \sum_{z \in Z} \boldsymbol{P}(Y, z)$
- Conditioning on Z:
  - $\boldsymbol{P}(Y) = \sum_{z \in Z} \boldsymbol{P}(Y|z)\boldsymbol{P}(z)$ (discrete)
  - $\boldsymbol{P}(Y) = \int \boldsymbol{P}(Y|z)\boldsymbol{P}(z)dz$ (continuous)
    $= \mathbb{E}_{\boldsymbol{z} \sim P(z)}\mathrm{P(Y|z)}$ (expected value notation)
- Bayes' Rule
  $$P(H|D) = \frac{P(D|H) \cdot P(H)}{P(D)} = \frac{P(D|H) \cdot P(H)}{\sum_h P(D|h)P(h)}$$

Color Convention in this Course

- Formulae, when occurring inline
- Newly introduced terminology and definitions
- Important results (observations, theorems) as well as emphasizing some aspects
- Examples are given with standard orange with possibly light orange frame
- Comments and notes in nearly opaque post-it
- Algorithms and program code
- Reminders (in the grey fog of your memory)

Today's lecture is based on the following

- A. Vergari, Y. Choi, R. Peharz, G. van den Broeck: Probabilistic Circuits, Tutorial at AAAI 2020, pp.1 – 80, http://starai.cs.ucla.edu/slides/AAAI20.pdf

References

- R. Dechter, K. Kask, and R. Mateescu. Iterative Join-Graph Propagation. arXiv e-prints, page arXiv:1301.0564, Dec. 2012.
- A. Kulesza and F. C. Pereira. Structured learning with approximate inference. In NIPS, 2007.
- Dagum, Luby: Approximating probabilistic inference in Bayesian belief networks is NP-hard, In Artificial Intelligence 60.1, pp. 141-153, 1993.
- J. D. Park and A. Darwiche. Complexity results and approximation strategies for map explanations. J. Artif. Int. Res., 21(1):101–133, Feb. 2004.
- C. de Campos. New complexity results for map in bayesian networks. In IJCAI 2011, vol. 11, pp. 2100–2106
- U. Oztok, A. Choi, and A. Darwiche. Solving pppp-complete problems using knowledge compilation. In Proceedings of KR'16, pp 94–103, 2016.
- P. Khosravi, Y. Liang, Y. Choi, and G. Van den Broeck. What to Expect of Classifiers? Reasoning about Logistic Regression with Missing Features. arXiv e-prints, page arXiv:1903.01620, Mar. 2019.
- N. Cohen, O. Sharir, and A. Shashua. On the Expressive Power of Deep Learning: A Tensor Analysis. arXiv e-prints, page arXiv:1509.05009, Sept. 2015.
- I. Kobyzev, S. J. D. Prince, and M. A. Brubaker. Normalizing Flows: An Introduction and Review of Current Methods. arXiv e-prints, page arXiv:1908.09257, Aug. 2019.
- A. Darwiche and P. Marquis. A Knowledge Compilation Map. arXiv e-prints, page arXiv:1106.1819, June 2011.
- D. Conaty, D. D. Maua, and C. P. de Campos. Approximation Complexity of Maximum A Posteriori Inference in Sum-Product Networks. arXiv e-prints, page arXiv:1703.06045, Mar. 2017.
- R. Gens and D. Pedro. Learning the structure of sum-product networks. In, volume 28 of Proceedings of Machine Learning Research, pages 873–880, 2013
- R. Peharz, A. Vergari, K. Stelzner, A. Molina, X. Shao, M. Trapp, K. Kersting, and Z. Ghahramani. Random sum-product networks: A simple and effective approach to probabilistic deep learning. In volume 115 of Proceedings of Machine Learning Research, pages 334–344, 2020.