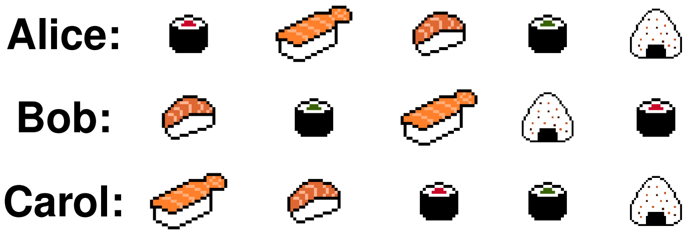
Two Perspectives to Learning with Circuits

Motivation

Given a selection of sushi...

...and people's preferences...



...how can we model this as a probability distribution...

$p(1^{\text{st}} =$ $, 3^{\text{rd}} =$ $)$

$p(2^{\text{nd}} =$ $| 1^{\text{st}} =$ $)$

$\arg\max p(1^{\text{st}} = ?, 2^{\text{nd}} = ?, 3^{\text{rd}} = ?, 4^{\text{th}} =$ $, 5^{\text{th}} =$ $)$

$p((3^{\text{rd}} =$ $\rightarrow 1^{\text{st}} =$ $) \vee 2^{\text{nd}} =$ $)$

...and extract meaningful queries from it?

Kamishima [2003]

Motivation

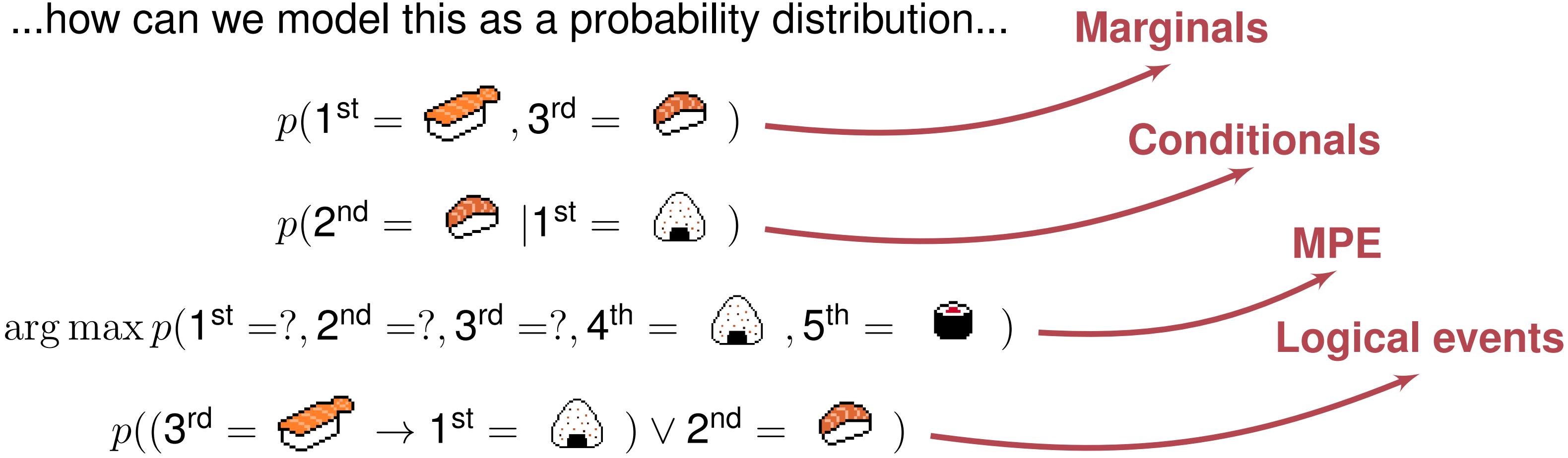
Given a selection of sushi...

🍙 🍤 🔴 🍣 🟢

...and people's preferences...

Alice:	🔴	🍤	🍣	🟢	🍙
Bob:	🍣	🟢	🍤	🍙	🔴
Carol:	🍤	🍣	🔴	🟢	🍙

...how can we model this as a probability distribution...



p(1st = 🍤 , 3rd = 🍣)

p(2nd = 🍣 |1st = 🍙)

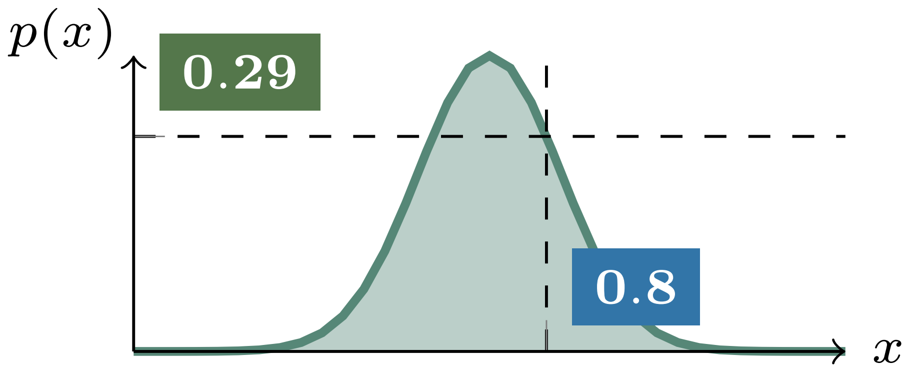
arg max *p*(1st =?, 2nd =?, 3rd =?, 4th = 🍙 , 5th = 🔴)

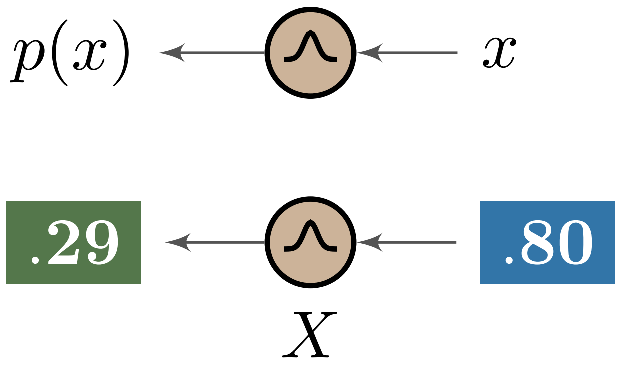
p((3rd = 🍤 → 1st = 🍙) ∨ 2nd = 🍣)

...and extract meaningful queries from it?

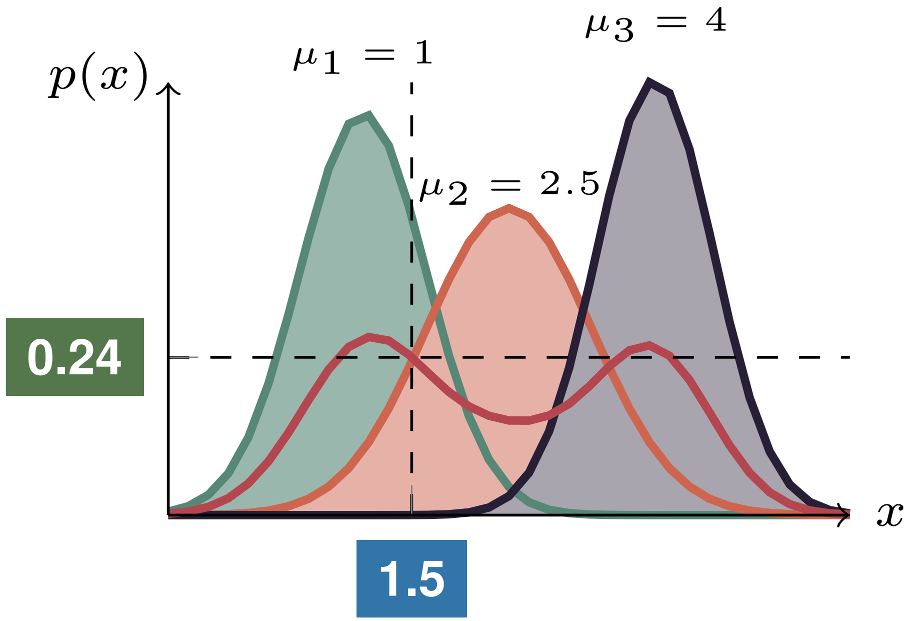
Kamishima [2003]

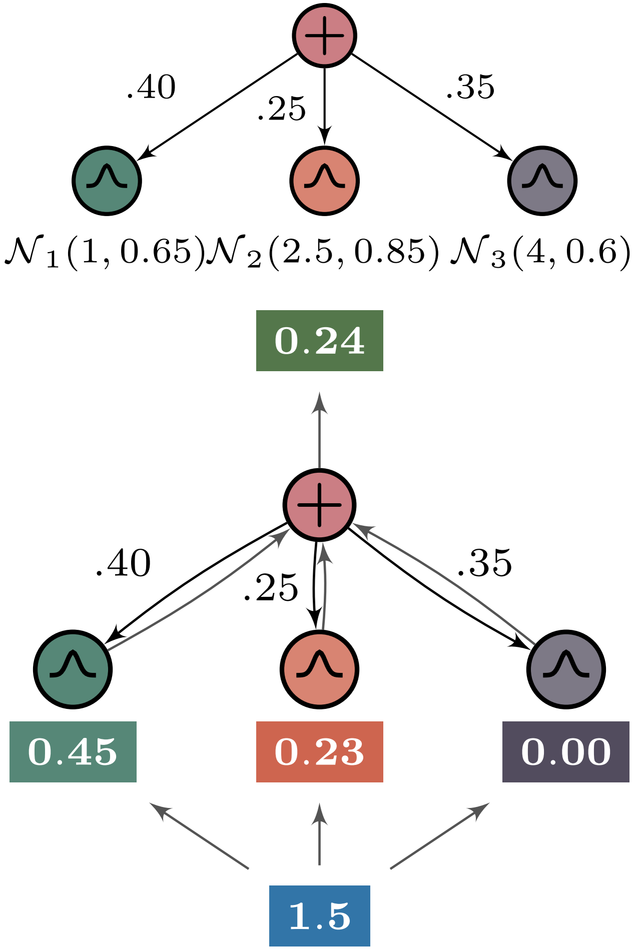
Probabilistic Circuits – Inputs



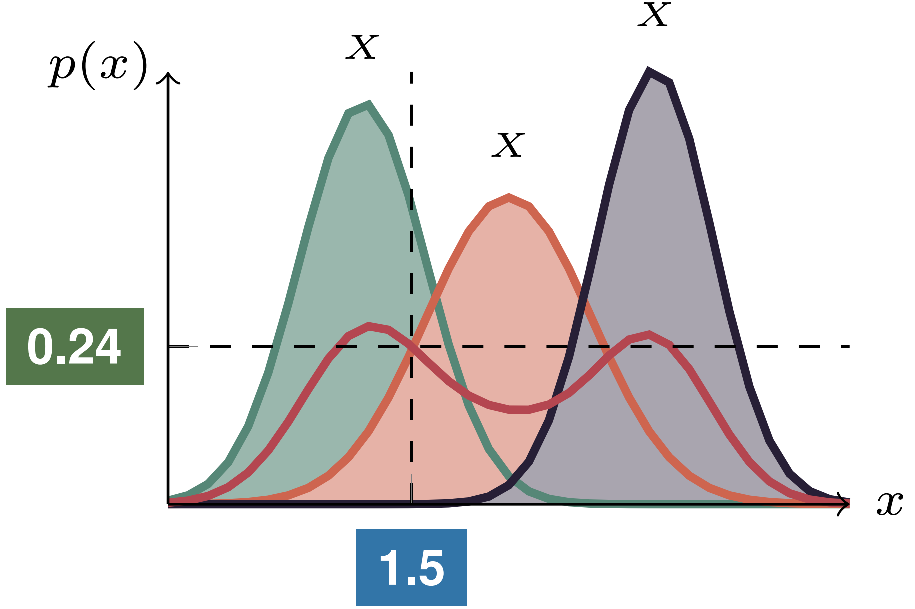


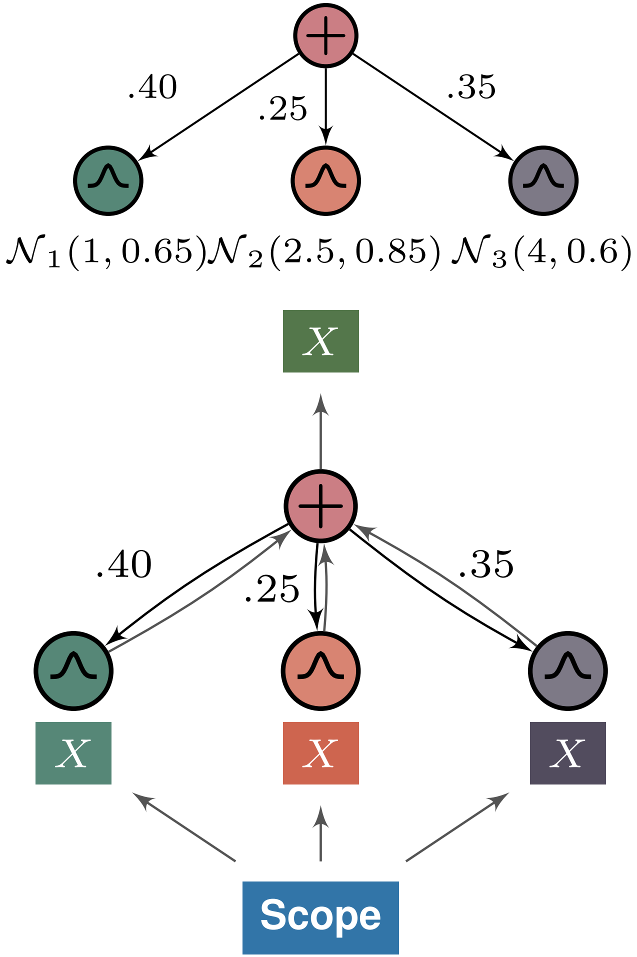
Probabilistic Circuits – Sums





Probabilistic Circuits – Smoothness

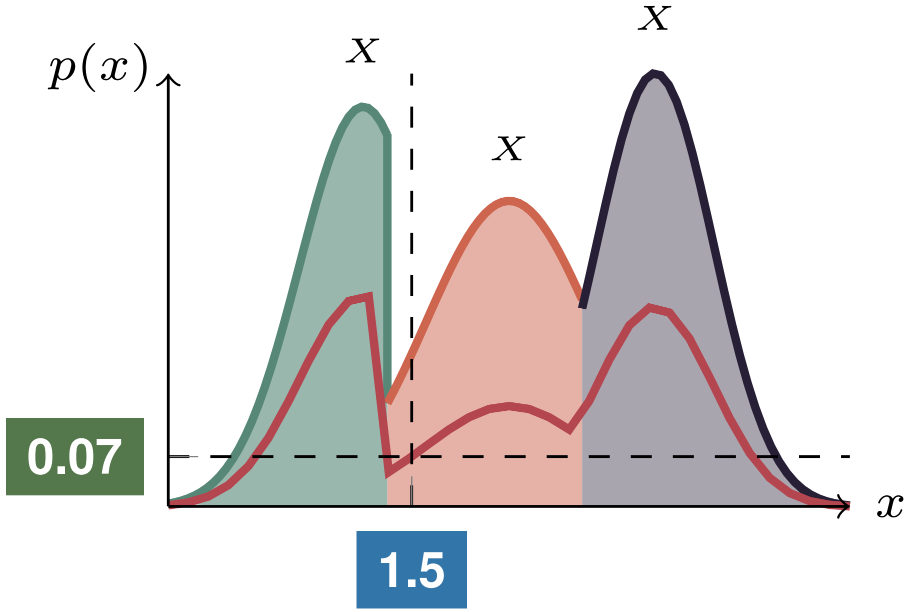


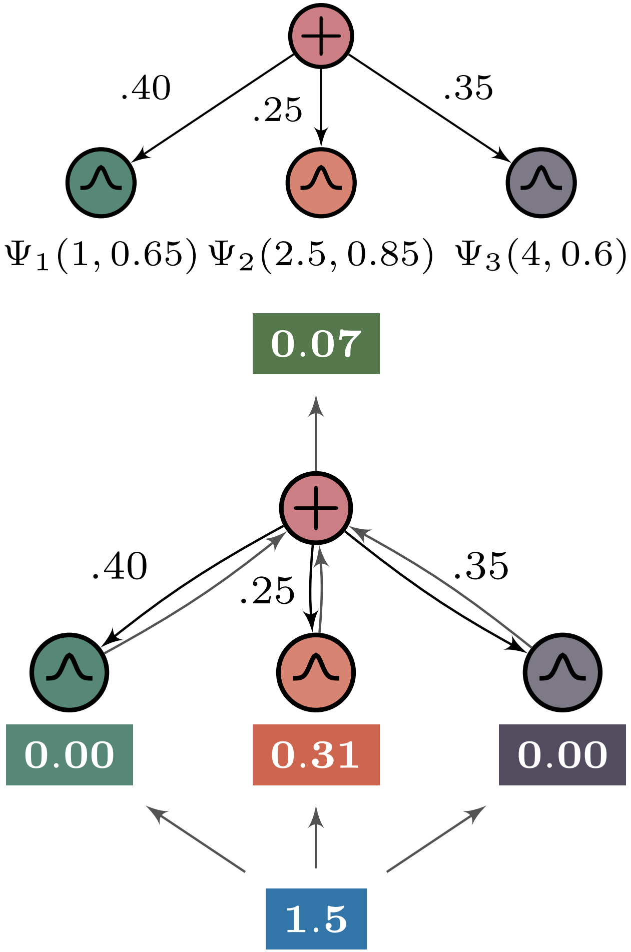


Definition 1 (Smoothness)**.**
Every sum node child mentions the <u>*same*</u> *variables.*

Darwiche [2001a]

Probabilistic Circuits – Determinism

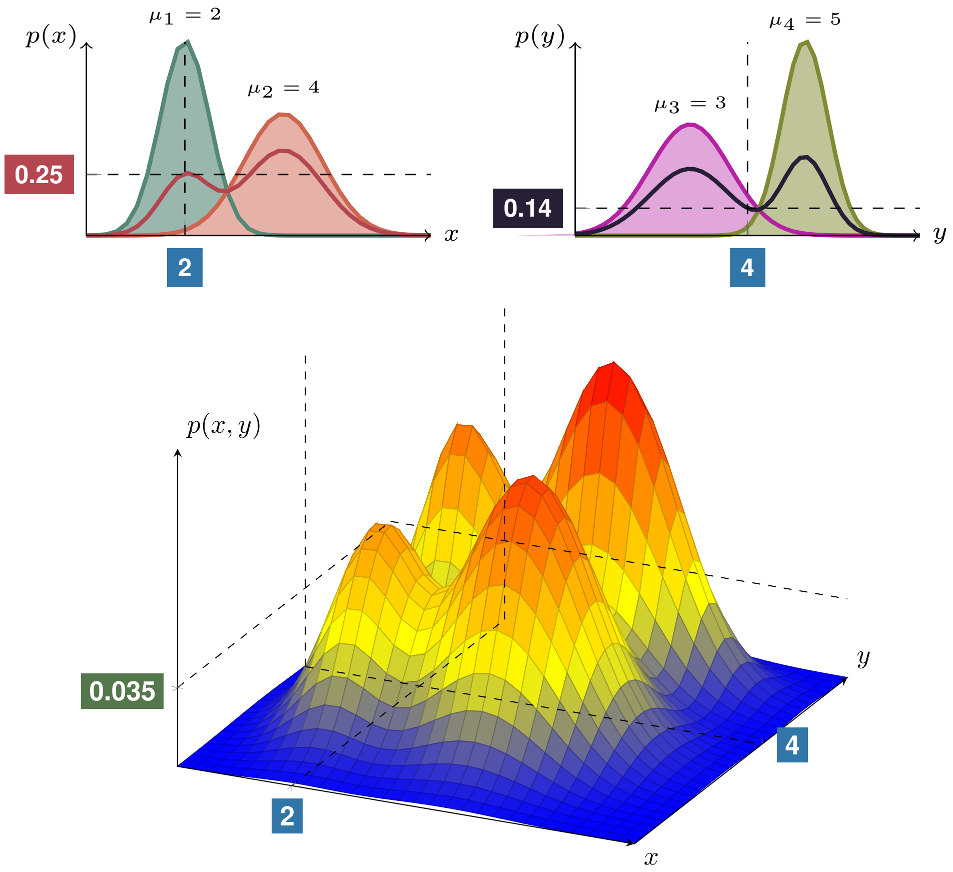


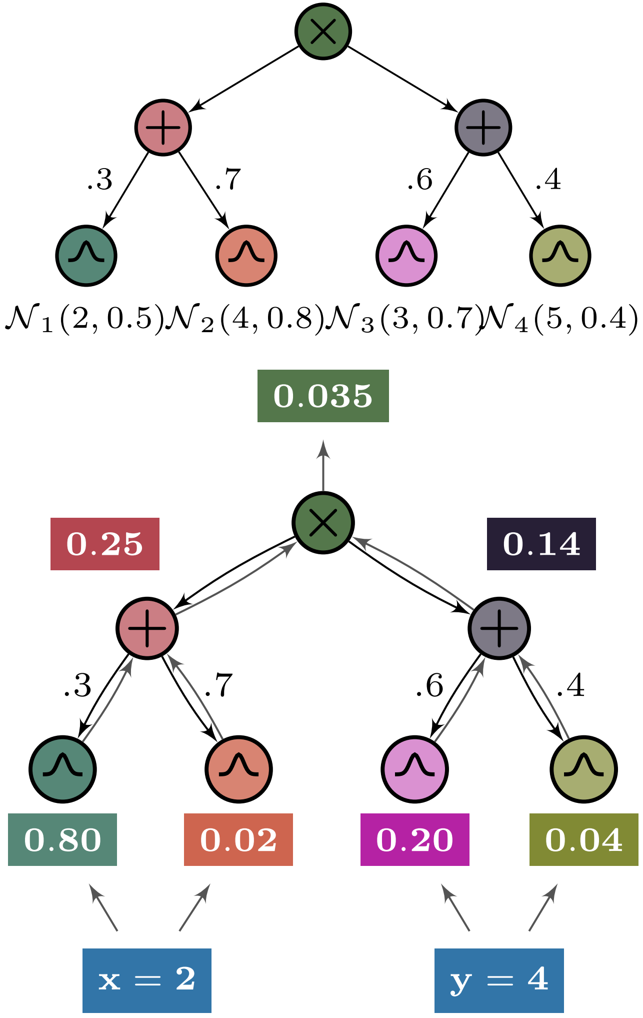


Definition 2 (Determinism)**.**
<u>At most one</u> sum node child has a positive value.

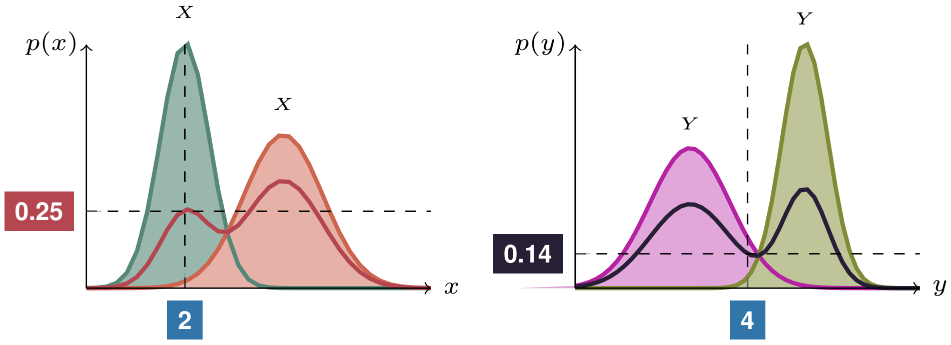
Darwiche [2001a]

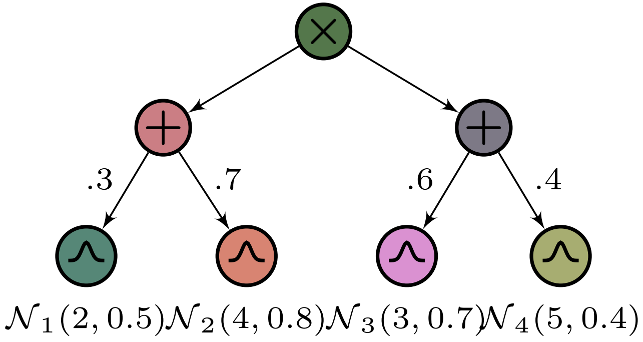
Probabilistic Circuits – Products

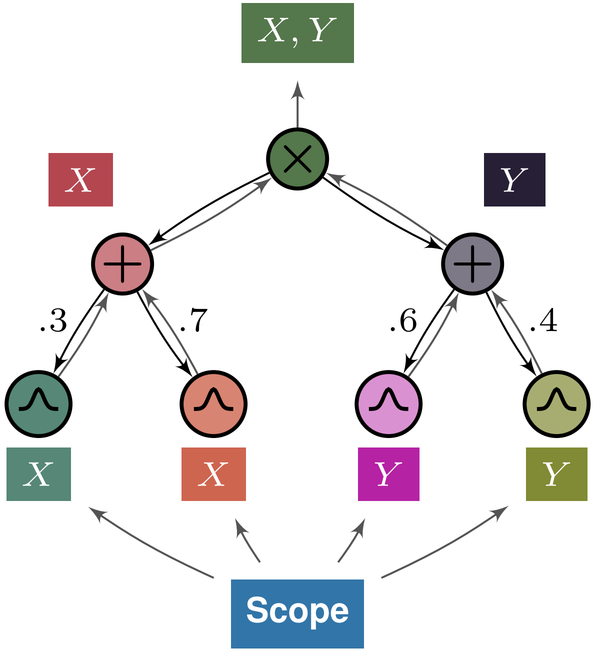




Probabilistic Circuits – Decomposability



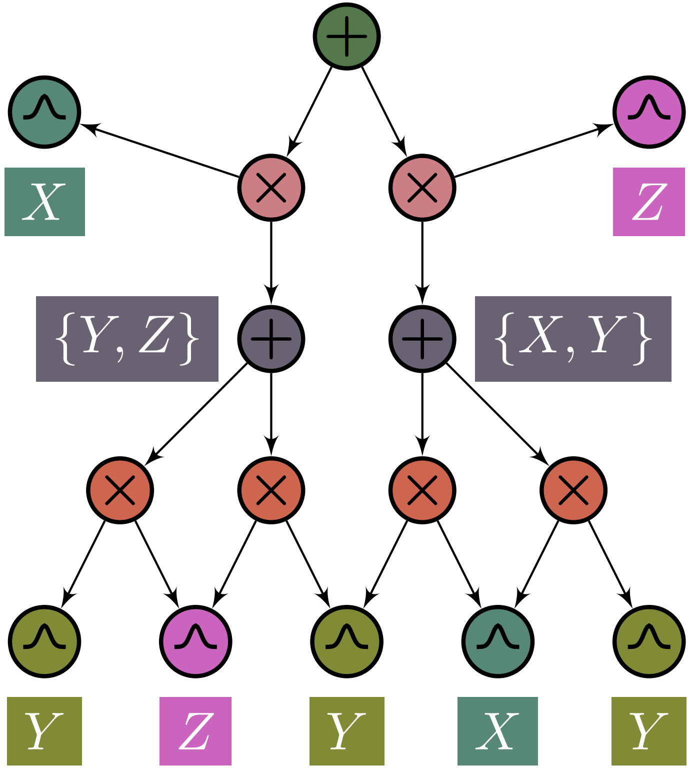


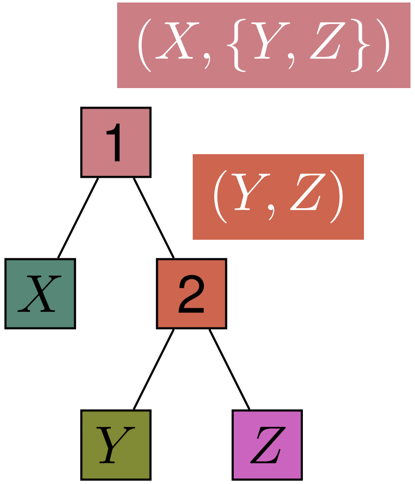


Definition 3 (Decomposability)**.**
Every product node child mentions different variables.

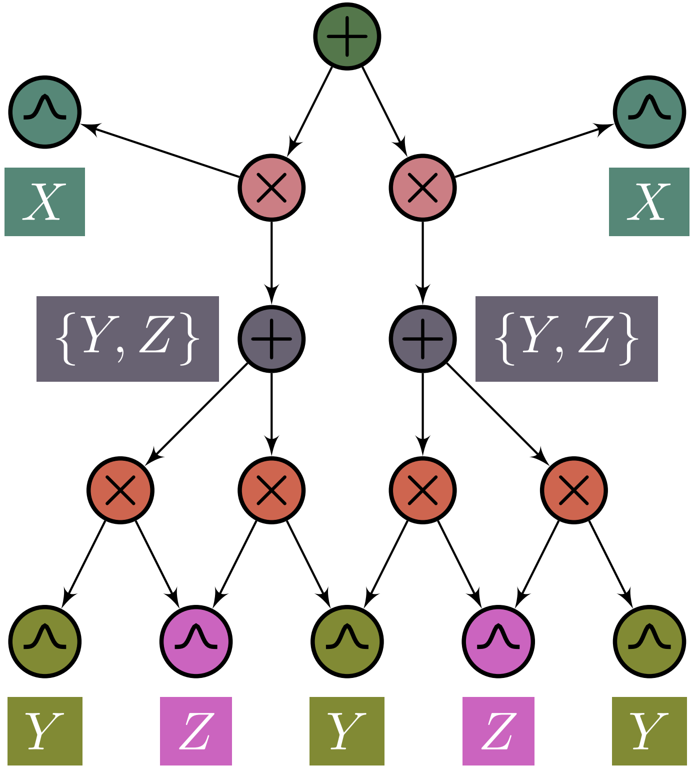
Darwiche [1999, 2001b]

Probabilistic Circuits – Structured Decomposability





vtree



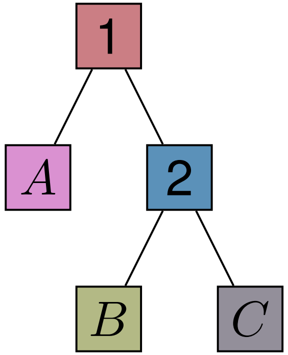
Definition 4 (Structured decomposability)**.** *Every product node follows a vtree decomposition.*

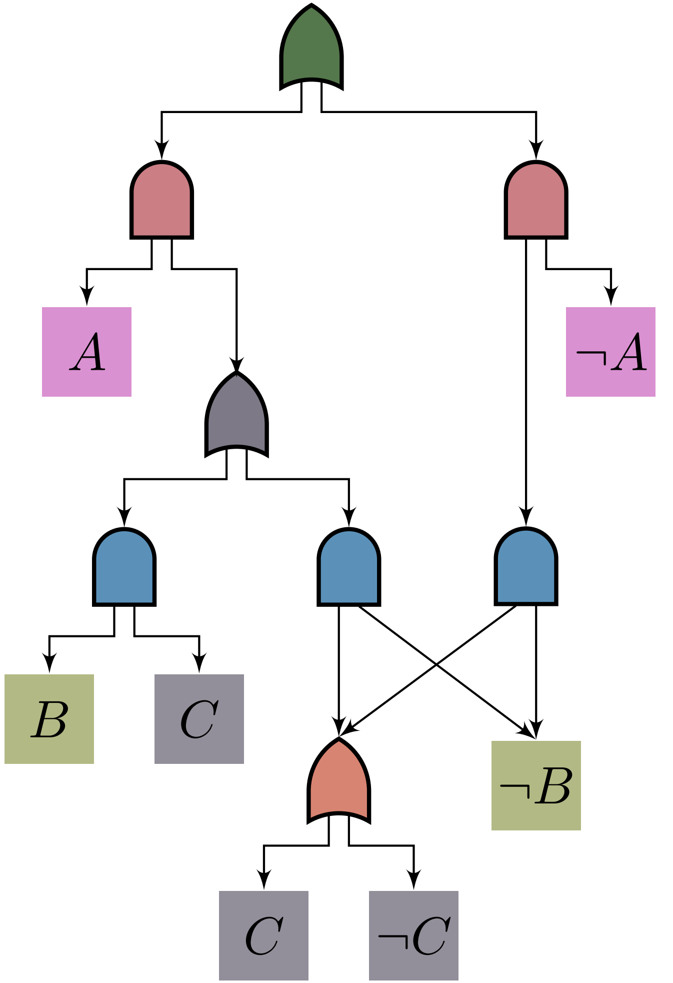
Pipatsrisawat and Darwiche [2008]

Probabilistic Circuits – Logic Circuits

A	B	C	$\phi(\mathbf{x})$
0	0	0	1
1	0	0	1
0	1	0	0
1	1	0	0
0	0	1	1
1	0	1	1
0	1	1	0
1	1	1	1

$$\phi(A, B, C) = (A \vee B) \wedge (\neg B \vee C)$$



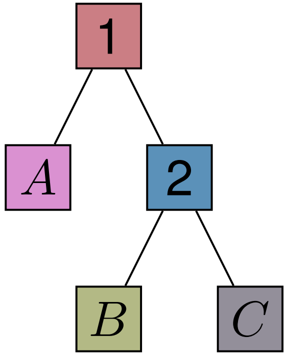


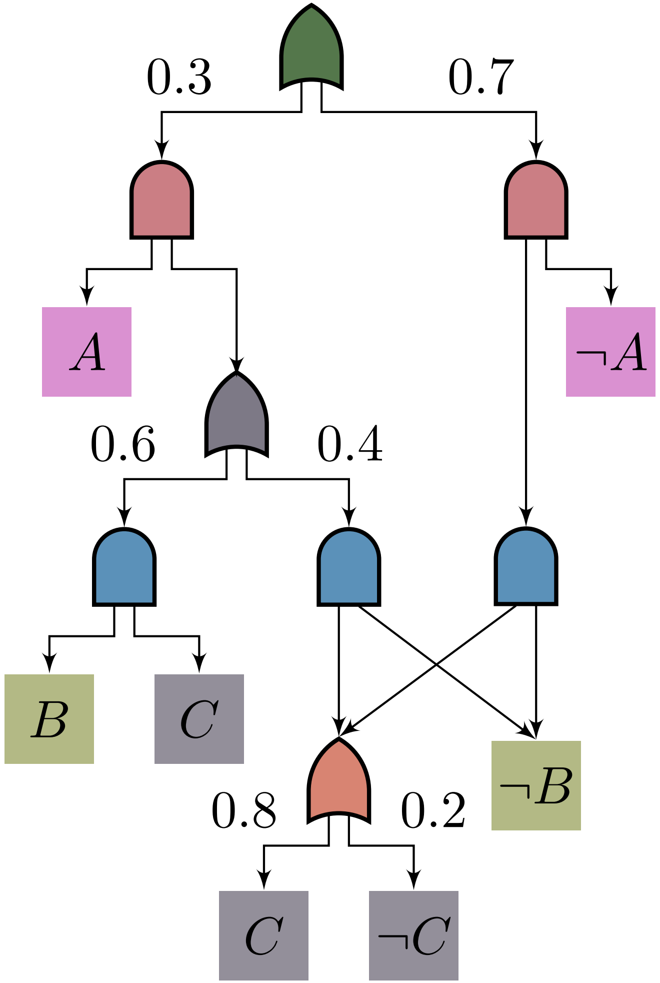
Darwiche [2011]

Probabilistic Circuits – Support

A	B	C	$\phi(\mathbf{x})$	$p(\mathbf{x})$
0	0	0	1	0.140
1	0	0	1	0.024
0	1	0	0	0.000
1	1	0	0	0.000
0	0	1	1	0.560
1	0	1	1	0.096
0	1	1	0	0.000
1	1	1	1	0.180

$$\phi(A, B, C) = (A \vee B) \wedge (\neg B \vee C)$$





Kisa et al. [2014]

Probabilistic Circuits – Tractability

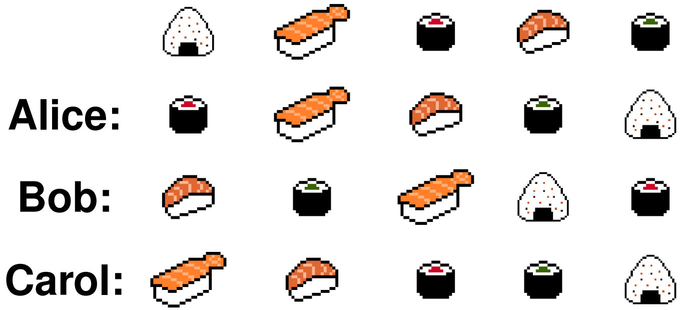
Query	+Sm?	+Dec?	+Det?	+Str Dec?
Evidence	✓	✓	✓	✓
Marginals	✗	✓	✓	✓
Conditionals	✗	✓	✓	✓
MPE	✗	✗	✓	✓
Shannon Entropy	✗	✗	✓	✓
Rényi Entropy	✗	✗	✓	✓
Cross Entropy	✗	✗	✗	✓
Kullback-Leibler Div	✗	✗	✗	✓
Rényi’s Alpha Div	✗	✗	✗	✓
Cauchy-Schwarz Div	✗	✗	✗	✓
Logical Events	✗	✗	✗	✓
Mutual Information	✗	✗	✗	✓

Vergari et al. [2021], Poon and Domingos [2011], Peharz et al. [2016]

Learning Probabilistic Circuits – Where are we right now?

Name	Class	Time Complexity	# hyperparams	Accepts logic?	Sm?	Dec?	Det?	Str Dec?	$\{0,1\}$?	$\mathbb{N}$?	$\mathbb{R}$?	Reference
LearnSPN	DIV	$\mathcal{O}(nkmc)$, if sum $\mathcal{O}(nm^3)$, if product	≥ 2	✗	✓	✓	✗	✗	✓	✓	✓	Gens and Domingos [2013]
ID-SPN	DIV	$\mathcal{O}(nkmc)$, if sum $\mathcal{O}(nm^3)$, if product $\mathcal{O}(ic(rn+m))$, if input	$\geq 2+3$	✗	✓	✓	✗	✗	✓	✓	✗	Rooshenas and Lowd [2014]
Prometheus	DIV	$\mathcal{O}(nkmc)$, if sum $\mathcal{O}(m(\log m)^2)$, if product	≥ 1	✗	✓	✓	✗	✗	✓	✓	✓	Jaini et al. [2018]
LearnPSDD	INCR	$\mathcal{O}(m^2)$, top-down vtree $\mathcal{O}(m^4)$, bottom-up vtree $\mathcal{O}(i\lvert\mathcal{C}\rvert^2)$, circuit structure	1	✓	✓	✓	✓	✓	✓	✗	✗	Liang et al. [2017]
Strudel	INCR	$\mathcal{O}(m^2 n)$, CLT + vtree $\mathcal{O}(i(\lvert\mathcal{C}\rvert n+m^2))$, circuit structure	1	✓	✓	✓	✓	✓	✓	✗	✗	Dang et al. [2020]
RAT-SPN	RAND	$\mathcal{O}(rd(s+l))$	4	✗	✓	✓	✗	✗	✓	✓	✓	Peharz et al. [2020]
XPC	RAND	$\mathcal{O}(i(t+kn)+ikm^2n)$	3	✗	✓	✓	✓	✓	✓	✗	✗	Mauro et al. [2021]
SamplePSDD	RAND	$\mathcal{O}(m)$, random vtree $\mathcal{O}(kc\log c+\log_2^2 k)$, per call	1	✓	✓	✓	✓	✓	✓	✗	✗	Geh and Mauá [2021]
LearnRP	RAND	$\mathcal{O}(m^2)$, top-down vtree $\mathcal{O}(m^4)$, bottom-up vtree $\mathcal{O}(knm)$, per call	0	✗	✓	✓	✗	✓	✓	✓	✓	To appear

SamplePSDD – Why?



If we assume

n sushi types,

k sized rankings with $k \leq n$,

X_{ij} binary variables; i is sushi type, j is position in ranking;

then the total number of possible assignments of the $n \cdot k$ variables is 2^{nk}...

...but many of these are zero probability assignments!

If we can embed total ranking constraints...

...we go down to k! total assignments!

Takeaway: models which exploit domain knowledge are much more efficient!

Example:

$n = 3,\ k = 3$

X_{11}	X_{12}	X_{13}	X_{21}	$\cdots$	X_{33}	$p(\mathbf{x}) > 0$
0	0	0	0	0	0	0
1	0	0	0	0	0	0
0	1	0	0	0	0	0
1	1	0	0	0	0	0
0	0	1	0	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
0	1	1	1	1	1	0
1	1	1	1	1	1	0

Assignments: $2^{3 \cdot 3} = 512$

Positive assignments: $3! = 6$

Choi et al. [2015]

SamplePSDD – Why?

Existing approaches:

LearnPSDD (Liang et al. [2017]):

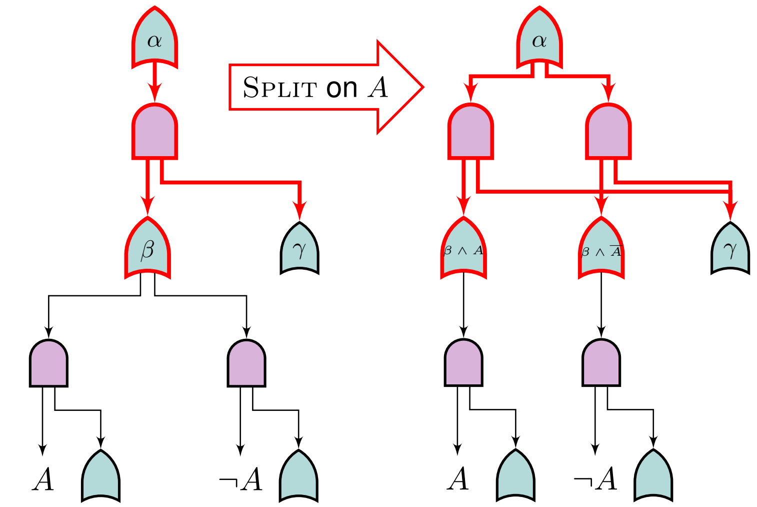
- ☒ Requires initial logic circuit encoding the support...
- ☒ Scales poorly to complex formulae and/or high dimension...
- ☒ Costly whole circuit evaluation at every iteration...
- ☑ Very good performance!

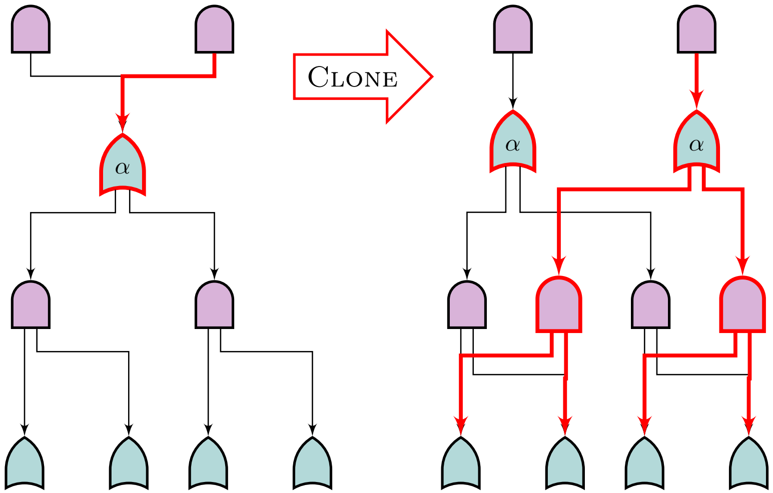
Strudel (Dang et al. [2020]):

- ☑ Constructs an initial structure (from a CLT)!
- ☒ But does not encode constraints...
- ☑ Scales to high dimension!
- ☒ As long as the circuit doesn't get too big...

SamplePSDD (Geh and Mauá [2021]):

- ☑ Scales to high dimension and complex formulae!
- ☑ Constructs a structure consistent with constraints!
- ☒ But does so by relaxing the formula...
- ☒ Performance varies on set bounds and vtree structure...



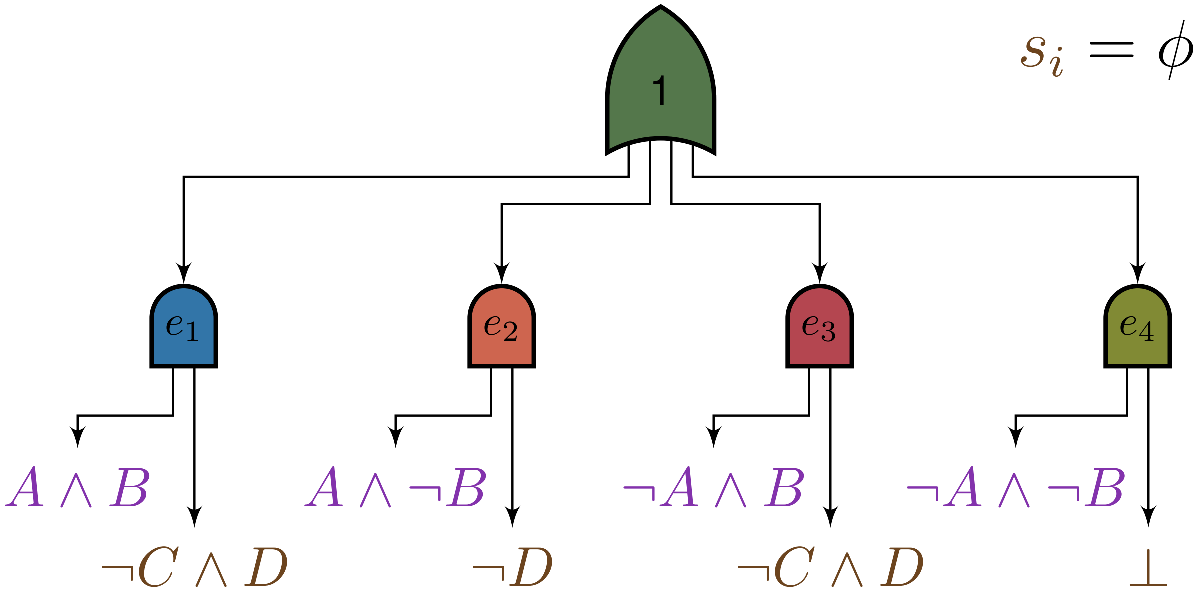


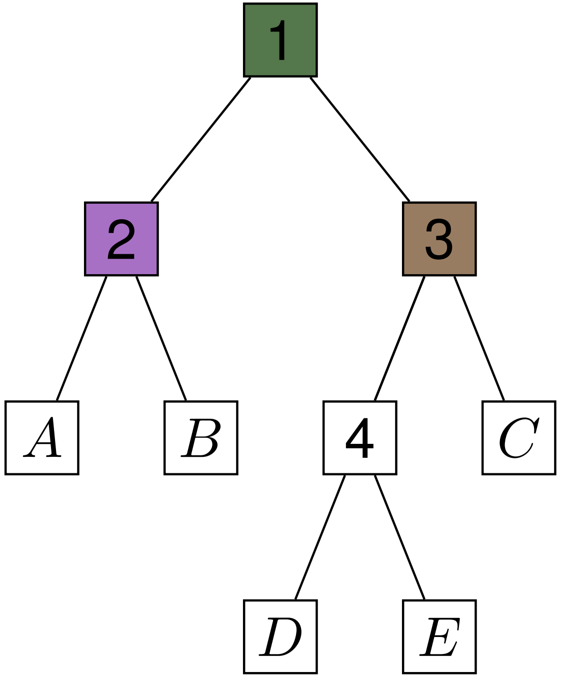
SamplePSDD

Common assumption: p_i are conjunctions of literals.

$$\phi(A,B,C,D) = (A \wedge \neg B \wedge \neg D) \vee (B \wedge \neg C \wedge D)$$

$$s_i = \phi|_{p_i}$$



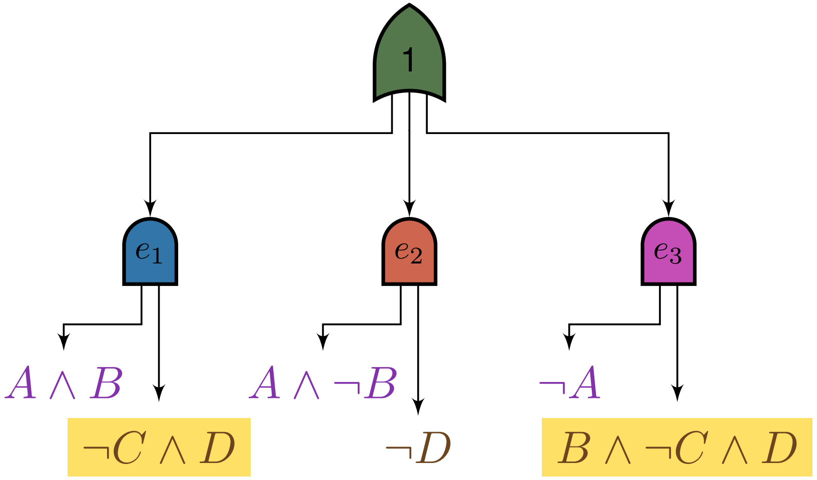


Problem: size of circuit is exponential in the size of p_i's scope.

SamplePSDD

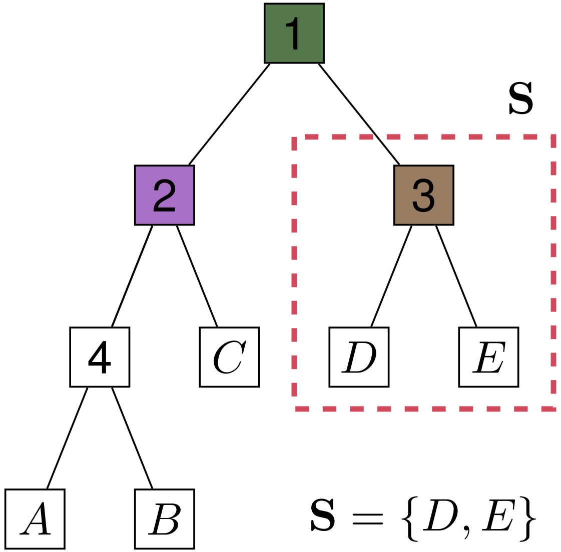
Solution: randomly sample a bounded number (k) of p_i

$$\phi(A, B, C, D) = (A \wedge \neg B \wedge \neg D) \vee (B \wedge \neg C \wedge D)$$



$$s_i = \phi|_{p_i}$$

$\mathrm{Sc}(s_3) \not\subseteq \mathbf{S}$



$\mathbf{S} = \{D, E\}$

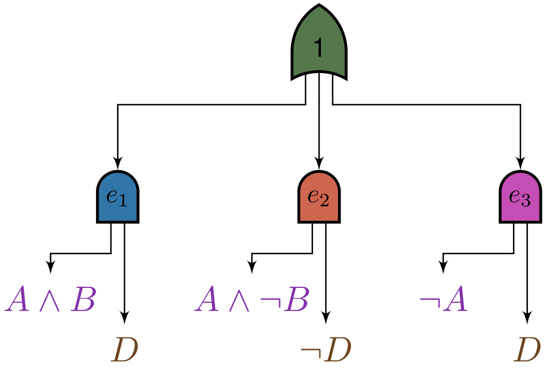
But: this violates structured decomposability:

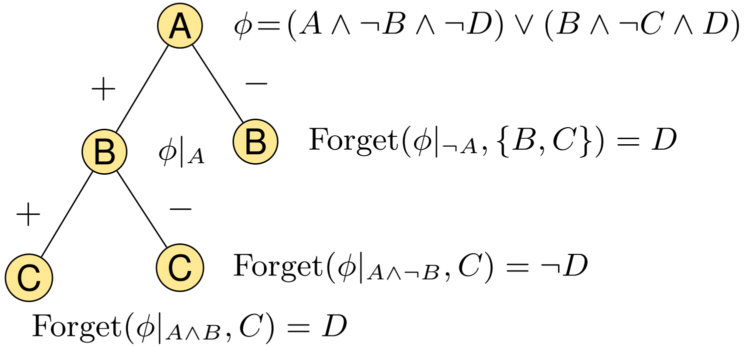
$\neg C \wedge D$ contains C, and $C \notin \mathbf{S}$

$\neg B \wedge \neg C \wedge D$ contains B and C, and $B, C \notin \mathbf{S}$

SamplePSDD

New solution: relax logical constraints ϕ





$\phi = (A \wedge \neg B \wedge \neg D) \vee (B \wedge \neg C \wedge D)$

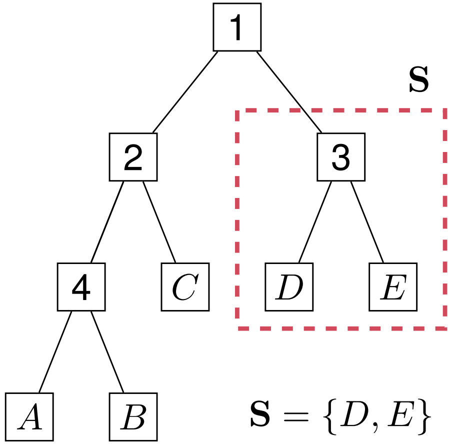
$\phi|_A$

$\text{Forget}(\phi|_{\neg A}, \{B, C\}) = D$

$\text{Forget}(\phi|_{A \wedge \neg B}, C) = \neg D$

$\text{Forget}(\phi|_{A \wedge B}, C) = D$

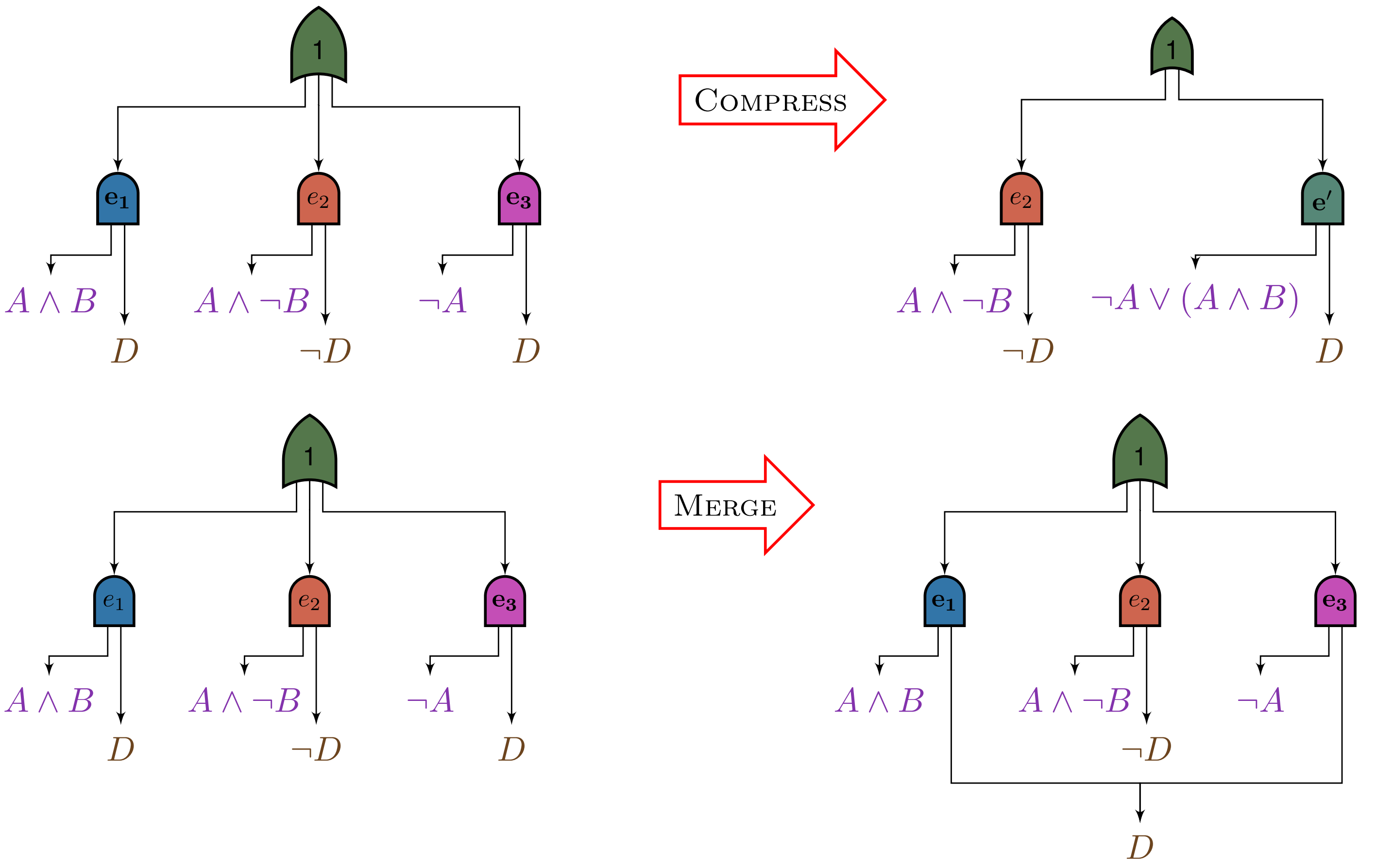
Now all s_i respect S



$\mathbf{S} = \{D, E\}$

SamplePSDD

Apply **local transformations** for variety and size reduction



Darwiche [2011]

SamplePSDD – Experiments

Evaluation: we sample 30 PSDDs and use 5 ensemble strategies:

- ● **Likelihood weighting (LLW)**,
- ■ **Uniform weights**,
- ◆ **Expectation Maximization (EM)**,
- ▲ **Stacking**,
- ▼ **Bayesian Model Combination (BMC)**;

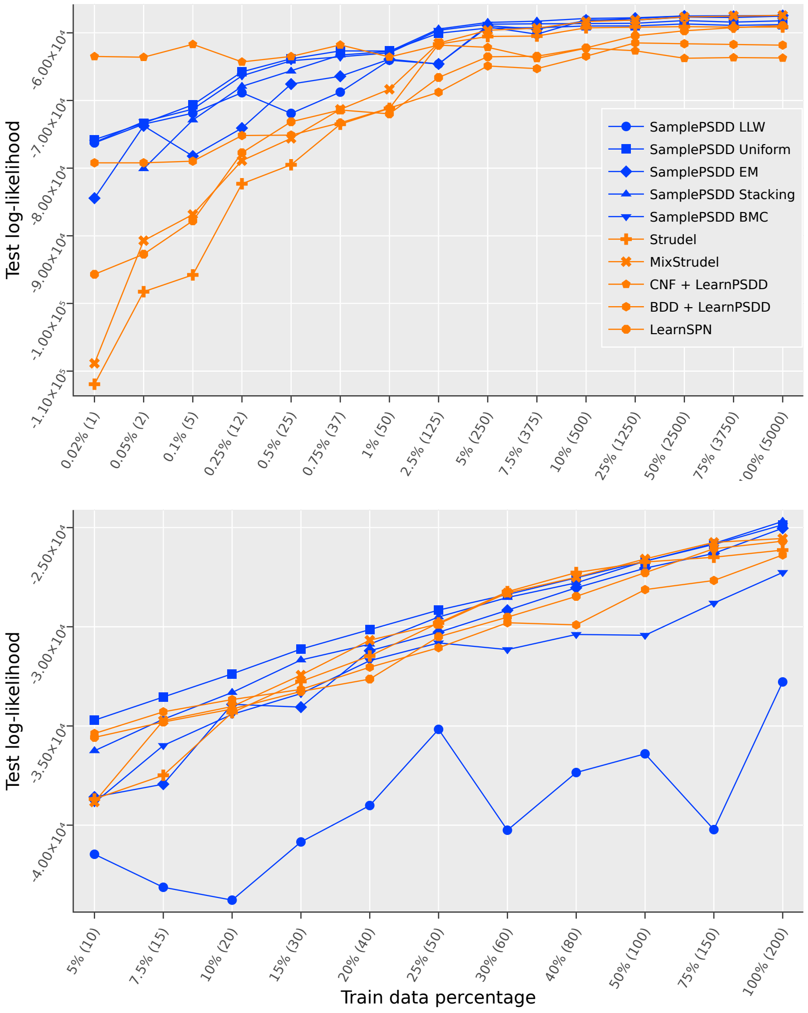
comparing against **Strudel**, **LearnPSDD** and **LearnSPN**.

Datasets: we evaluate with 5 data + knowledge as logic constraints:

	Dataset	#vars	#train	ϕ's size
⇒	LED	14	5000	23
⇒	LED + Images	157	700	39899
	Sushi Ranking	100	3500	17413
	Sushi Top 5	10	3500	37
	Dota 2 Games	227	92650	1308

Our approach fares **better** with **fewer** data, yet remains **competitive** under **lots of data**.

Mattei et al. [2020], Kamishima [2003], Shen et al. [2017], Choi et al. [2015], Gens and Domingos [2013], Dang et al. [2020]



SamplePSDD – Experiments

Evaluation: we sample 30 PSDDs and use 5 ensemble strategies:

- ● **Likelihood weighting (LLW)**,
- ■ **Uniform weights**,
- ◆ **Expectation Maximization (EM)**,
- ▲ **Stacking**,
- ▼ **Bayesian Model Combination (BMC)**;

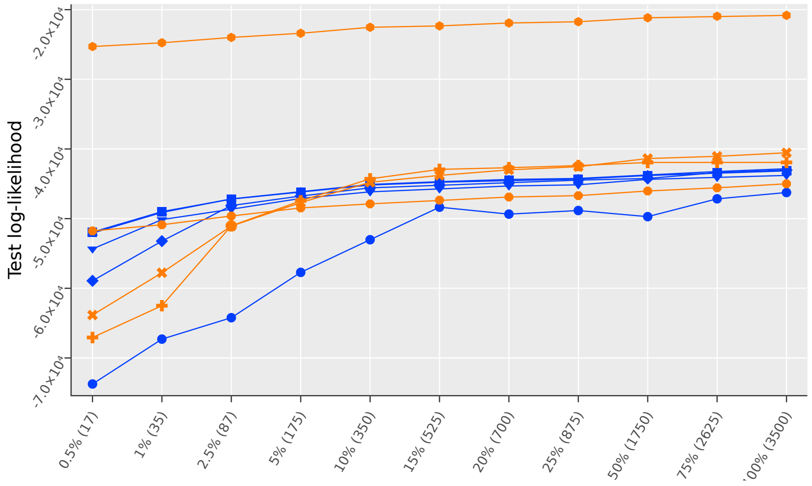
comparing against **Strudel**, **LearnPSDD** and **LearnSPN**.

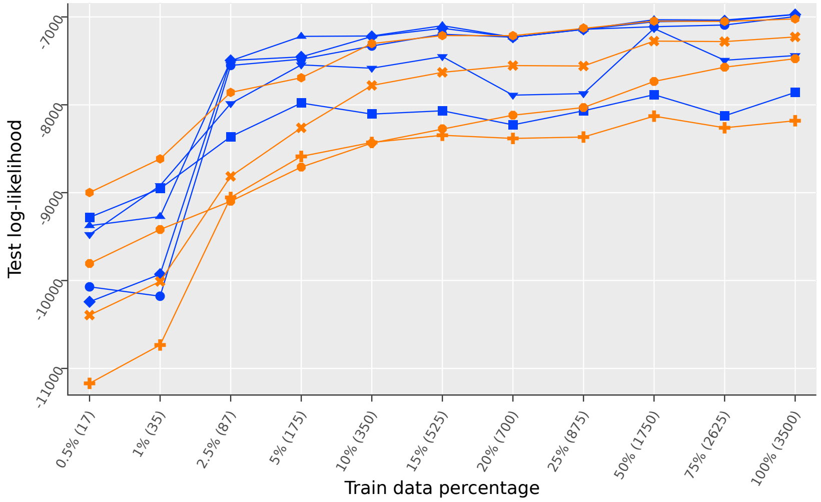
Datasets: we evaluate with 5 data + knowledge as logic constraints:

	Dataset	#vars	#train	ϕ's size
	LED	14	5000	23
	LED + Images	157	700	39899
⇒	Sushi Ranking	100	3500	17413
⇒	Sushi Top 5	10	3500	37
	Dota 2 Games	227	92650	1308

Our approach fares **better** with **fewer** data, yet remains **competitive** under **lots of data**.

Mattei et al. [2020], Kamishima [2003], Shen et al. [2017], Choi et al. [2015], Gens and Domingos [2013], Dang et al. [2020]





SamplePSDD – Experiments

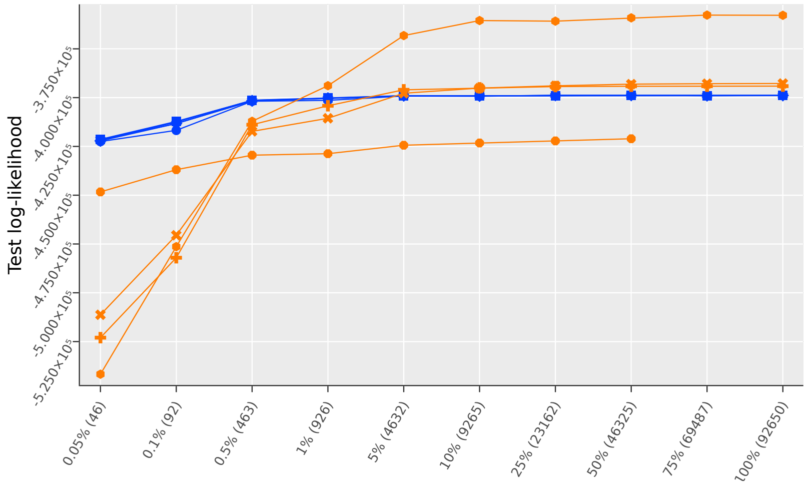
Evaluation: we sample 30 PSDDs and use 5 ensemble strategies:

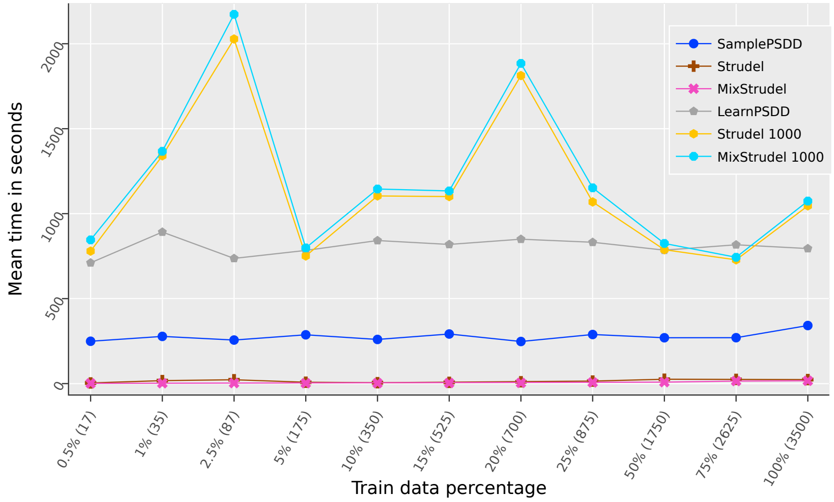
- ● **Likelihood weighting (LLW)**,
- ■ **Uniform weights**,
- ◆ **Expectation Maximization (EM)**,
- ▲ **Stacking**,
- ▼ **Bayesian Model Combination (BMC)**;

comparing against **Strudel**, **LearnPSDD** and **LearnSPN**.

Datasets: we evaluate with 5 data + knowledge as logic constraints:

	Dataset	#vars	#train	ϕ's size
	LED	14	5000	23
	LED + Images	157	700	39899
	Sushi Ranking	100	3500	17413
	Sushi Top 5	10	3500	37
⇒	Dota 2 Games	227	92650	1308





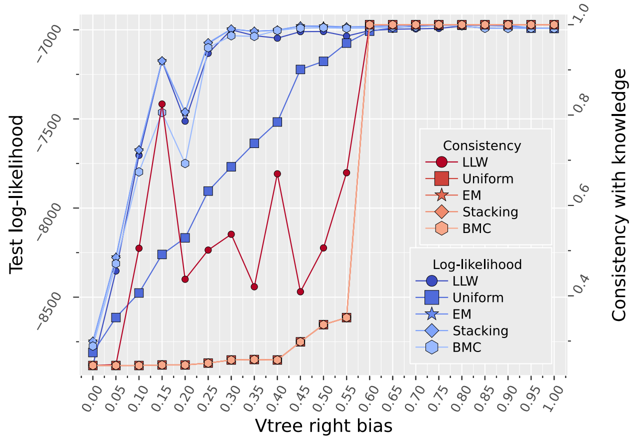
Our approach fares **better** with **fewer** data, yet remains **competitive** under **lots of data**.

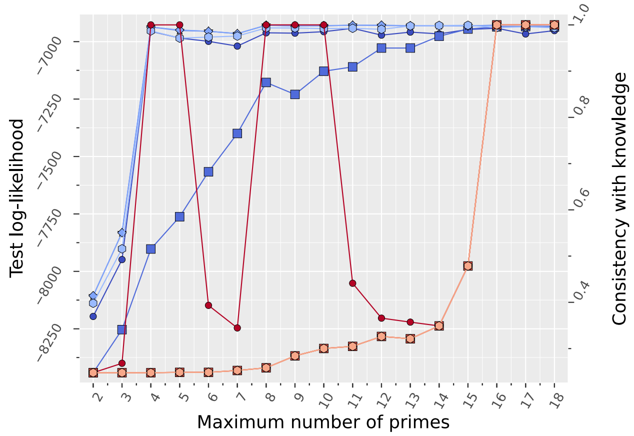
Mattei et al. [2020], Kamishima [2003], Shen et al. [2017], Choi et al. [2015], Gens and Domingos [2013], Dang et al. [2020]

SamplePSDD – Experiments

What is the impact of higher k's and right-leaning vtrees

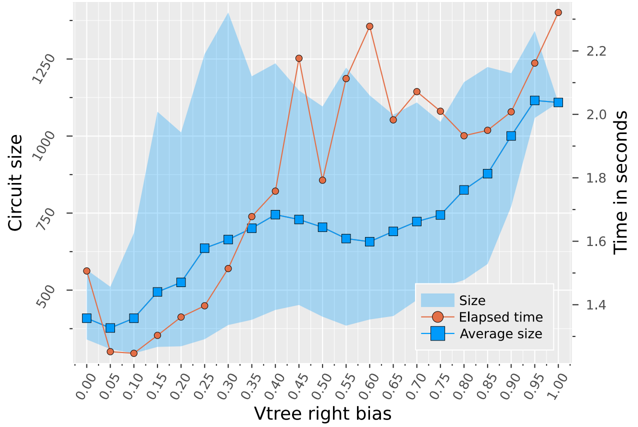
in log-likelihood and consistency?

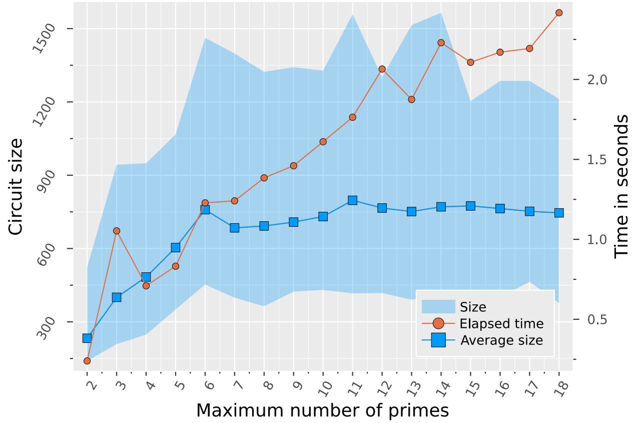




Samples perform **better** with higher k's and right-leaning vtrees...

...but at a **cost** to complexity.


Circuit size
Time in seconds
Vtree right bias
Size
Elapsed time
Average size


Circuit size
Time in seconds
Maximum number of primes
Size
Elapsed time
Average size