Tractable Probabilistic Models

Guy Van den Broeck
University of California, Los Angeles

based on a joint UAI-19 tutorial with

Antonio Vergari
University of California, Los Angeles

Nicola Di Mauro
University of Bari

<table>
<thead>
<tr>
<th>AoGs</th>
<th>PDGs</th>
<th>N Bs</th>
<th>Fully factorized</th>
<th>sd-DNNF</th>
<th>PS DDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trees</td>
<td>L TMs</td>
<td>DNNFs</td>
<td>OBDDs</td>
<td>C Nets</td>
<td>SPNs</td>
</tr>
<tr>
<td>Thin Junction Trees</td>
<td>NNF</td>
<td>FBDDs</td>
<td>BDDs</td>
<td>ACs</td>
<td>VAEs</td>
</tr>
<tr>
<td>Polytrees</td>
<td>d-NNFs</td>
<td>ADDs</td>
<td>SDDs</td>
<td>TACs</td>
<td>GANs</td>
</tr>
<tr>
<td>Mixtures</td>
<td>XADDs</td>
<td>XSDDs</td>
<td>MNs</td>
<td>BNs</td>
<td>FGs</td>
</tr>
</tbody>
</table>

The Alphabet Soup of models in AI
Logical and Probabilistic models
Tractable and Intractable probabilistic models
Expressive models without compromises
Why tractable inference?
or expressiveness vs tractability

Probabilistic circuits
a unified framework for tractable models
Why tractable inference?
or expressiveness vs tractability

Probabilistic circuits
a unified framework for tractable models

Building circuits
learning them from data and compiling other models

Applications
what are circuits useful for
Tractable Probabilistic Circuits @ ICLP?

- Logical roots of probabilistic circuits
- Probabilistic circuits bridge between logic and deep learning
- Bring back world models!
- Powerful general reasoning tool
  
  \[\Rightarrow\]  
  \textit{for example in probabilistic logic programming}

- Elegant knowledge representation formalism
Why tractable inference?

or the inherent trade-off of tractability vs. expressiveness
Why probabilistic inference?

**q₁**: What is the probability that today is a Monday and there is a traffic jam on Herzl Str.?

**q₂**: Which day is most likely to have a traffic jam on my route to work?
Why probabilistic inference?

q₁: What is the probability that today is a Monday and there is a traffic jam on Herzl Str.?

q₂: Which day is most likely to have a traffic jam on my route to work?

⇒ fitting a predictive model!
Why probabilistic inference?

$q_1$: What is the probability that today is a Monday and there is a traffic jam on Herzl Str.?

$q_2$: Which day is most likely to have a traffic jam on my route to work?

$\implies$ fitting a predictive model!

$\implies$ answering probabilistic queries on a probabilistic model of the world $m$

$q_1(m) = ?$  $q_2(m) = ?$
Why probabilistic inference?

$q_1$: What is the probability that today is a Monday and there is a traffic jam on Herzl Str.?

$X = \{\text{Day, Time, Jam}_{\text{Str1}}, \text{Jam}_{\text{Str2}}, \ldots, \text{Jam}_{\text{StrN}}\}$

$q_1(m) = p_m(\text{Day} = \text{Mon}, \text{Jam}_{\text{Herzl}} = 1)$
Why probabilistic inference?

$q_1$: What is the probability that today is a Monday and there is a traffic jam on Herzl Str.?

$X = \{\text{Day, Time, Jam}_{\text{Str1}}, \text{Jam}_{\text{Str2}}, \ldots, \text{Jam}_{\text{StrN}}\}$

$q_1(m) = p_m(\text{Day = Mon, Jam}_{\text{Herzl}} = 1)$

$\Rightarrow$ marginals
Why probabilistic inference?

$q_2$: Which day is most likely to have a traffic jam on my route to work?

$X = \{ \text{Day, Time, } \text{Jam}_{\text{Str1}}, \text{Jam}_{\text{Str2}}, \ldots, \text{Jam}_{\text{StrN}} \}$

$q_2(m) = \arg\max_d p_m(\text{Day} = d \land \bigvee_{i \in \text{route}} \text{Jam}_{\text{Str}i})$
Why probabilistic inference?

$q_2$: Which day is most likely to have a traffic jam on my route to work?

$X = \{\text{Day}, \text{Time}, \text{Jam}_{\text{Str1}}, \text{Jam}_{\text{Str2}}, \ldots, \text{Jam}_{\text{StrN}}\}$

$q_2(m) = \arg\max_d p_m(\text{Day} = d \land \bigvee_{i \in \text{route}} \text{Jam}_{\text{Str} i})$

$\Rightarrow$ marginals + MAP + logical events
Tractable Probabilistic Inference

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

$\Rightarrow$ often poly will in fact be linear!
Tractable Probabilistic 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 $O(\text{poly}(|q| \cdot |m|))$.

$\Rightarrow$ often poly will in fact be *linear*!

Note: if $M$ and $Q$ are compact in the number of random variables $X$, that is, $|m|, |q| \in O(\text{poly}(|X|))$, then query time is $O(\text{poly}(|X|))$. 
What about approximate inference?

- Why approximate when we can do exact? 
  - and do we lose something in terms of expressiveness?

- Approximations can be intractable as well [Dagum et al. 1993; Roth 1996]
  - But sometimes approximate inference comes with guarantees

- Approximate inference by exact inference in approximate model 
  [Dechter et al. 2002; Choi et al. 2010; Lowd et al. 2010; Sontag et al. 2011; Friedman et al. 2018]

- Approximate inference (even with guarantees) can mislead learners 
  [Kulesza et al. 2007]
  - Chaining approximations is flying with a blindfold on
What about approximate inference?

- Why approximate when we can do exact?
  
  ⇒ and do we lose something in terms of expressiveness?

- Approximations can be intractable as well [Dagum et al. 1993; Roth 1996]
  
  ⇒ But sometimes approximate inference comes with guarantees

- Approximate inference by exact inference in approximate model
  
  [Dechter et al. 2002; Choi et al. 2010; Lowd et al. 2010; Sontag et al. 2011; Friedman et al. 2018]

- Approximate inference (even with guarantees) can mislead learners
  
  [Kulesza et al. 2007]
  
  ⇒ Chaining approximations is flying with a blindfold on
What about approximate inference?

- Why approximate when we can do exact? and do we lose something in terms of expressiveness?
- Approximations can be intractable as well [Dagum et al. 1993; Roth 1996] But sometimes approximate inference comes with guarantees
- Approximate inference by exact inference in approximate model [Dechter et al. 2002; Choi et al. 2010; Lowd et al. 2010; Sontag et al. 2011; Friedman et al. 2018]
- Approximate inference (even with guarantees) can mislead learners [Kulesza et al. 2007] Chaining approximations is flying with a blindfold on
What about approximate inference?

Why approximate when we can do exact?
⇒ and do we lose something in terms of expressiveness?

Approximations can be intractable as well [Dagum et al. 1993; Roth 1996]
⇒ But sometimes approximate inference comes with guarantees

Approximate inference by exact inference in approximate model
[Dechter et al. 2002; Choi et al. 2010; Lowd et al. 2010; Sontag et al. 2011; Friedman et al. 2018]

Approximate inference (even with guarantees) can mislead learners
[Kulesza et al. 2007]
⇒ Chaining approximations is flying with a blindfold on
Stay Tuned For ...

Next:
1. What are classes of queries?
2. Are my favorite models tractable?
3. Are tractable models expressive?

After: We introduce probabilistic circuits as a unified framework for tractable probabilistic modeling
**Complete evidence queries (EVI)**

$q_3$: *What is the probability that today is a Monday at 12.00 and there is a traffic jam only on Herzl Str.?*
**Complete evidence queries (EVI)**

$q_3$: What is the probability that today is a Monday at 12.00 and there is a traffic jam only on Herzl Str.?

$X = \{\text{Day, Time, Jam}_{\text{Herzl}}, \text{Jam}_{\text{Str2}}, \ldots, \text{Jam}_{\text{StrN}}\}$

$q_3(m) = p_m(X = \{\text{Mon, 12.00, 1, 0, \ldots, 0}\})$
**Complete evidence queries (EVI)**

**q₃:** *What is the probability that today is a Monday at 12.00 and there is a traffic jam only on Herzl Str.?*

\[ X = \{ \text{Day, Time, Jam}_{\text{Herzl}}, \text{Jam}_{\text{Str2}}, \ldots, \text{Jam}_{\text{StrN}} \} \]

\[ q₃(m) = p_m(X = \{ \text{Mon, 12.00, 1, 0, \ldots, 0} \}) \]

...fundamental in *maximum likelihood learning*

\[ \theta_{m}^{\text{MLE}} = \arg\max_{\theta} \prod_{x \in D} p_{m}(x; \theta) \]
Generative Adversarial Networks

$$\min_{\theta} \max_{\phi} \mathbb{E}_{x \sim p_{\text{data}}(x)} \left[ \log D_{\phi}(x) \right] + \mathbb{E}_{z \sim p(z)} \left[ \log (1 - D_{\phi}(G_{\theta}(z))) \right]$$

Goodfellow et al., “Generative adversarial nets”, 2014
Generative Adversarial Networks

$$\min_{\theta} \max_{\phi} \mathbb{E}_{x \sim p_{\text{data}}(x)} \left[ \log D_{\phi}(x) \right] + \mathbb{E}_{z \sim p(z)} \left[ \log(1 - D_{\phi}(G_{\theta}(z))) \right]$$

- no explicit likelihood!
  $\Rightarrow$ adversarial training instead of MLE
  $\Rightarrow$ no tractable EVI
- good sample quality
  $\Rightarrow$ but lots of samples needed for MC
- unstable training
  $\Rightarrow$ mode collapse

Goodfellow et al., “Generative adversarial nets”, 2014
**Variational Autoencoders**

\[ p_{\theta}(x) = \int \, p_{\theta}(x \mid z)p(z) \, dz \]

- an explicit likelihood model!

---

*Rezende et al., “Stochastic backprop. and approximate inference in deep generative models”, 2014*

*Kingma et al., “Auto-Encoding Variational Bayes”, 2014*
Variational Autoencoders

\[
\log p_\theta(x) \geq \mathbb{E}_{z \sim q_\phi(z|x)} \left[ \log p_\theta(x \mid z) \right] - \text{KL}(q_\phi(z \mid x) \mid \mid p(z))
\]

- an explicit likelihood model!
- ... but computing \( \log p_\theta(x) \) is intractable
  \[\Rightarrow\] an infinite and uncountable mixture
  \[\Rightarrow\] no tractable EVI
- we need to optimize the ELBO...
  \[\Rightarrow\] which is “broken”

[Alemi et al. 2017; Dai et al. 2019]
**Probabilistic Graphical Models (PGMs)**

*Declarative semantics*: a clean separation of modeling assumptions from inference

**Nodes**: random variables  
**Edges**: dependencies

**Inference**:  
- conditioning [Darwiche 2001; Sang et al. 2005]  
- elimination [Zhang et al. 1994; Dechter 1998]  
- message passing [Yedidia et al. 2001; Dechter et al. 2002; Choi et al. 2010; Sontag et al. 2011]
Markov Networks (MNs)

\[ p(X) = \frac{1}{Z} \prod_c \phi_c(X_c) \]
Markov Networks (MNs)

\[ p(X) = \frac{1}{Z} \prod_c \phi_c(X_c) \]

\[ Z = \int \prod_c \phi_c(X_c) dX \]

\[ \Rightarrow \quad \text{EVI queries are intractable!} \]
**PGMs: MNs and BNs**

**Markov Networks (MNs)**

\[
p(X) = \frac{1}{Z} \prod_c \phi_c(X_c)
\]

\[
Z = \int \prod_c \phi_c(X_c) dX
\]

⇒ EVI queries are intractable!

**Bayesian Networks (BNs)**

\[
p(X) = \prod_i p(X_i | pa(X_i))
\]

⇒ EVI queries are tractable!
Marginal queries (MAR)

$q_1$: What is the probability that today is a Monday at 12:00 and there is a traffic jam only on Herzl Str.?
Marginal queries (MAR)

$q_1$: What is the probability that today is a Monday at 12:00 and there is a traffic jam only on Herzl Str.?

$q_1(m) = p_m(Day = Mon, Jam_{Herzl} = 1)$
**Marginal queries (MAR)**

$q_1$: What is the probability that today is a Monday at 12:00 and there is a traffic jam only on Herzl Str.?

$q_1(m) = p_m(\text{Day} = \text{Mon}, \text{Jam}_{\text{Herzl}} = 1)$

General: $p_m(e) = \int p_m(e, H) \, dH$

where $E \subset X$

$H = X \setminus E$

pinterest.com/pin/190417890473268205/
Conditional queries (CON)

$q_4$: What is the probability that there is a traffic jam on Herzl Str. given that today is a Monday?
Conditional queries (CON)

$q_4$: What is the probability that there is a traffic jam on Herzl Str. given that today is a Monday?

$q_4(m) = p_m(Jam_{Herzl} = 1 \mid Day = Mon)$
Conditional queries (CON)

$q_4$: *What is the probability that there is a traffic jam on Herzl Str. given that today is a Monday?*

$q_4(m) = p_m(Jam_{Herzl} = 1 \mid Day = Mon)$

If you can answer MAR queries, then you can also do *conditional queries* (CON):

$$p_m(Q \mid E) = \frac{p_m(Q, E)}{p_m(E)}$$
Complexity of MAR on PGMs

**Exact complexity:** Computing MAR and COND is \#P-complete [Cooper 1990; Roth 1996].

**Approximation complexity:** Computing MAR and COND approximately within a relative error of $2^{n^{1-\epsilon}}$ for any fixed $\epsilon$ is NP-hard [Dagum et al. 1993; Roth 1996].

**Treewidth:** Informally, how tree-like is the graphical model $m$?
Formally, the minimum width of any tree-decomposition of $m$.

**Fixed-parameter tractable:** MAR and CON on a graphical model $m$ with treewidth $w$ take time $O(|X| \cdot 2^w)$, which is linear for fixed width $w$ [Dechter 1998; Koller et al. 2009].

⇒ what about bounding the treewidth by design?
**Complexity of MAR on PGMs**

*Exact complexity:* Computing MAR and COND is \textit{\#P-complete} [Cooper 1990; Roth 1996].

*Approximation complexity:* Computing MAR and COND approximately within a relative error of $2^{n^{1-\epsilon}}$ for any fixed $\epsilon$ is \textit{NP-hard} [Dagum et al. 1993; Roth 1996].

*Treewidth:* Informally, how tree-like is the graphical model $m$? Formally, the minimum width of any tree-decomposition of $m$.

*Fixed-parameter tractable:* MAR and CON on a graphical model $m$ with treewidth $w$ take time $O(|X| \cdot 2^w)$, which is linear for fixed width $w$ [Dechter 1998; Koller et al. 2009].

⇒ what about bounding the treewidth by design?
Low-treewidth PGMs

Trees
[Meilă et al. 2000]

Polytrees
[Dasgupta 1999]

Thin Junction trees
[Bach et al. 2001]

If treewidth is bounded (e.g. $\approx 20$), exact MAR and CON inference is possible in practice
**Low-treewidth PGMs: trees**

A tree-structured BN [Meilä et al. 2000] where each $X_i \in \mathbf{X}$ has at most one parent $\text{Pa}_{X_i}$.

\[
p(\mathbf{X}) = \prod_{i=1}^{n} p(x_i | \text{Pa}_{x_i})
\]

**Exact querying:** EVI, MAR, CON tasks linear for trees: $O(|\mathbf{X}|)$

**Exact learning** from $d$ examples takes $O(|\mathbf{X}|^2 \cdot d)$ with the classical Chow-Liu algorithm\(^1\)

---

\(^1\)Chow et al., “Approximating discrete probability distributions with dependence trees”, 1968
What do we lose?

**Expressiveness**: Ability to compactly represent rich and complex classes of distributions

Bounded-treewidth PGMs lose the ability to represent *all possible distributions* ...

---

Martens et al., “On the Expressive Efficiency of Sum Product Networks”, 2014
Mixtures as a convex combination of $k$ (simpler) probabilistic models

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

EVI, MAR, CON queries scale linearly in $k$
Mixtures

Mixtures as a convex combination of \( k \) (simpler) probabilistic models

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

Mixtures are marginalizing a **categorical latent variable** \( Z \) with \( k \) values

\( \Rightarrow \) increased expressiveness
Expressiveness and efficiency

**Expressiveness**: Ability to compactly represent rich and effective classes of functions

⇒ *mixture of Gaussians can approximate any distribution!*

**Expressive efficiency (succinctness)** compares model sizes in terms of their ability to compactly represent functions

⇒ *but how many components do they need?*

---

Martens et al., “On the Expressive Efficiency of Sum Product Networks”, 2014
Mixture models

Expressive efficiency

⇒ deeper mixtures would be efficient compared to shallow ones
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?
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?

$q_5(m) = \arg\max_j p_m(j_1, j_2, \ldots | \text{Day} = M, \text{Time} = 9)$
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?

$q_5(m) = \arg\max_j p_m(j_1, j_2, \ldots | \text{Day} = M, \text{Time} = 9)$

General: $\arg\max_q p_m(q | e)$

where $Q \cup E = X$
**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!

$$\max_q p_m(q \mid e) = \max_q \sum_z p_m(q, z \mid e)$$

$$\neq \sum_z \max_q p_m(q, z \mid e)$$
**Marginal MAP (MMAP)**

aka Bayesian Network MAP

**Q6:** Which combination of roads is most likely to be jammed on Monday at 9am?
Marginal MAP (MMAP)
aka Bayesian Network MAP

$q_6$: Which combination of roads is most likely to be jammed on Monday at 9am?

$q_6(m) = \arg\max_j p_m(j_1, j_2, \ldots | \text{Time} = 9)$
Marginal MAP (MMAP) aka Bayesian Network MAP

$q_6$: Which combination of roads is most likely to be jammed on Monday at 9am?

$q_6 (m) = \text{argmax}_j \ p_m (j_1, j_2, \ldots | \text{Time} = 9)$

General: $\text{argmax}_q \ p_m (q | e) = \text{argmax}_q \ \sum_h p_m (q, h | e)$

where $Q \cup H \cup E = X$
Marginal MAP (MMAP)

aka Bayesian Network MAP

$q_6$: Which combination of roads is most likely to be jammed on Monday at 9am?

$q_6(m) = \arg\max_j p_m(j_1, j_2, \ldots | \text{Time} = 9)$

$\Rightarrow$ NP$^{PP}$-complete [Park et al. 2006]

$\Rightarrow$ NP-hard for trees [Campos 2011]

$\Rightarrow$ NP-hard even for Naive Bayes [ibid.]
q2: Which day is most likely to have a traffic jam on my route to work?
**Advanced queries**

$q_2$: Which day is most likely to have a traffic jam on my route to work?

$q_2(m) = \arg\max_d p_m(Day = d \land \bigvee_{i \in \text{route}} \text{JamStr}_i)$

$\Rightarrow$ **marginals + MAP + logical events**

---

**Advanced queries**

$q_2$: Which day is most likely to have a traffic jam on my route to work?

$q_7$: What is the probability of seeing more traffic jams in Jaffa than Marina?

*Bekker et al., “Tractable Learning for Complex Probability Queries”, 2015*
**Advanced queries**

$q_2$: Which day is most likely to have a traffic jam on my route to work?

$q_7$: What is the probability of seeing more traffic jams in Jaffa than Marina?

⇒ *counts + group comparison*

---

Advanced queries

q2: Which day is most likely to have a traffic jam on my route to work?

q7: What is the probability of seeing more traffic jams in Jaffa than Marina?

and more:

- expected classification agreement

- expected predictions [Khosravi et al. 2019a]

Fully factorized models

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 \textit{linear}!

\[ \Rightarrow \text{ but definitely not expressive...} \]
Expressive models are not very tractable...
and tractable ones are not very expressive...
probabilistic circuits are at the “sweet spot”
Probabilistic Circuits
1. What are the building blocks of tractable models?  
   ⇒ a computational graph forming a probabilistic circuit

2. For which queries are probabilistic circuits tractable?  
   ⇒ tractable classes induced by structural properties

After: How are probabilistic circuits related to the alphabet soup of models?
**Base Case: Univariate Distributions**

Generally, univariate distributions are tractable for:

- **EVI**: output $p(X_i)$ (density or mass)
- **MAR**: output 1 (normalized) or $Z$ (unnormalized)
- **MAP**: output the mode
Base Case: Univariate Distributions

\[
x \xrightarrow[\bigwedge]{X} p_X(x)
\]

Generally, univariate distributions are tractable for:

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

⇒ often 100% probability for one value of a categorical random variable

⇒ for example, \( X \) or \( \neg X \) for Boolean random variable
Generally, univariate distributions are tractable for:

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

⇒ often 100% probability for one value of a categorical random variable

⇒ for example, $X$ or $\neg X$ for Boolean random variable
Factorizations are products

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
Factorizations are products

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
Factorizations are products

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
Mixtures are sums

Also mixture models can be treated as a simple **computational unit** over distributions

\[
p(X) = w_1 \cdot p_1(X) + w_2 \cdot p_2(X)
\]
Mixtures are sums

Also mixture models can be treated as a simple computational unit over distributions

\[ p(x) = 0.2 \cdot p_1(x) + 0.8 \cdot p_2(x) \]
Mixtures are sums

Also mixture models can be treated as a simple computational unit over distributions

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

With mixtures, we increase expressiveness

⇒ by stacking them we increase expressive efficiency
A grammar for tractable models

Recursive semantics of probabilistic circuits

$\bigwedge_{X_1}$
A grammar for tractable models

Recursive semantics of probabilistic circuits
A grammar for tractable models

Recursive semantics of probabilistic circuits
A grammar for tractable models

Recursive semantics of probabilistic circuits
Probabilistic circuits are not PGMs!

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

<table>
<thead>
<tr>
<th>PGMs</th>
<th>Circuits</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Nodes:</strong> random variables</td>
<td>unit of computations</td>
</tr>
<tr>
<td><strong>Edges:</strong> dependencies</td>
<td>order of execution</td>
</tr>
<tr>
<td><strong>Inference:</strong></td>
<td></td>
</tr>
<tr>
<td>□ conditioning</td>
<td>□ feedforward pass</td>
</tr>
<tr>
<td>□ elimination</td>
<td>□ backward pass</td>
</tr>
<tr>
<td>□ message passing</td>
<td></td>
</tr>
</tbody>
</table>

⇒ they are computational graphs, more like neural networks
Just sum, products and distributions?

just arbitrarily compose them like a neural network!
Just sum, products and distributions?

Just arbitrarily compose them like a neural network!

⇒ structural constraints needed for tractability
How do we ensure tractability?
Decomposability

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

\[ X_1 \times X_2 \times X_3 \]

\[ \text{decomposable circuit} \]

\[ X_1 \times X_1 \times X_3 \]

\[ \text{non-decomposable circuit} \]

Darwiche et al., “A knowledge compilation map”, 2002
Smoothness
aka completeness

A sum node is smooth if its children depend of the same variable sets
⇒ otherwise not accounting for some variables

smooth circuit

⇒ smoothness can be easily enforced [Shih et al. 2019]

Darwiche et al., “A knowledge compilation map”, 2002
Tractable MAR/CON

Smoothness and decomposability enable tractable MAR/CON queries
Tractable MAR/CON

Smoothness and decomposability enable tractable MAR/CON queries

If \( p(x, y) = p(x)p(y) \), \textbf{(decomposability)}:

\[
\int \int p(x, y) \, dx \, dy = \int \int p(x)p(y) \, dx \, dy = \\
= \int p(x) \, dx \, \int p(y) \, dy
\]

\( \Rightarrow \text{larger integrals decompose into easier ones} \)
Tractable MAR/CON

Smoothness and decomposability enable tractable MAR/CON queries

If \( p(x) = \sum_i w_i p_i(x) \), (smoothness):

\[
\int p(x) dx = \int \sum_i w_i p_i(x) dx = \\
= \sum_i w_i \int p_i(x) dx
\]

\( \implies \) integrals are “pushed down” to children
Tractable MAR/CON

Smoothness and decomposability enable tractable MAR/CON queries

Forward pass evaluation for MAR

\[ \Rightarrow \text{linear in circuit size!} \]

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

- leafs over \( X_1 \) and \( X_3 \) output \( Z_i = \int p(x_i) \, dx_i \)

\[ \Rightarrow \text{for normalized leaf distributions: } 1.0 \]

- leafs over \( X_2 \) and \( X_4 \) output EVI
Determinism
aka selectivity

A sum node is deterministic if the output of only one children is non zero for any input

$\Rightarrow$ e.g. if their distributions have disjoint support

$X_1 \leq \theta \quad X_2 \quad X_1 > \theta \quad X_2$

$w_1 \quad w_2$

\textit{deterministic circuit}

\textit{non-deterministic circuit}
The addition of determinism enables tractable MAP queries!
The addition of determinism enables tractable MAP queries!

If $p(q, e) = p(q_x, e_x, q_y, e_y)$

$= p(q_x, e_x)p(q_y, e_y)$ (decomposable product node):

$$\arg\max_q p(q \mid e) = \arg\max_q p(q, e)$$

$$= \arg\max_{q_x, q_y} p(q_x, e_x, q_y, e_y)$$

$$= \arg\max_{q_x} p(q_x, e_x), \arg\max_{q_y} p(q_y, e_y)$$

$\implies$ solving optimization independently
The addition of determinism enables tractable MAP queries!

If \( p(q, e) = \sum_i w_i p_i(q, e) = w_c p_c(q, e) \),

(deterministic sum node):

\[
\text{argmax}_q p(q, e) = \text{argmax}_q \sum_i w_i p_i(q, e) \\
= \text{argmax}_q \max_i w_i p_i(q, e) \\
= \max_i \text{argmax}_q w_i p_i(q, e)
\]

\[\Rightarrow\] one non-zero child term, thus sum is max
The addition of determinism enables tractable MAP queries!

Evaluating the circuit twice: **bottom-up** and **top-down**  \(\Rightarrow\)  *still linear in circuit size!*
The addition of determinism enables tractable MAP queries!

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

In practice:
1. turn sum into max nodes
2. evaluate \(p(e)\) bottom-up
3. retrieve max activations top-down
4. compute MAP queries at leaves
The addition of determinism enables tractable MAP queries!

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

In practice:
1. turn sum into max nodes
2. evaluate \(p(e)\) bottom-up
3. retrieve max activations top-down
4. compute MAP queries at leaves
The addition of determinism enables tractable MAP queries!

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

In practice:

1. turn sum into max nodes
2. evaluate \(p(e)\) bottom-up
3. retrieve max activations top-down
4. compute MAP queries at leaves
The addition of determinism enables tractable MAP queries!

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

In practice:
1. turn sum into max nodes
2. evaluate $p(e)$ bottom-up
3. retrieve max activations top-down
4. compute MAP queries at leaves
Approximate MAP

If the probabilistic circuit is non-deterministic, MAP is intractable:

\[ \arg\max_q \sum_i w_i p_i(q, e) = \arg\max_q \sum_z p(q, z, e) \neq \arg\max_q \max_z p(q, z, e) \]

However, same two steps algorithm, still used as an approximation to MAP [Liu et al. 2013; Peharz et al. 2016]
Structured decomposability

A product node is structured decomposable if decomposes according to a node in a \textit{vtree} ⇒ stronger requirement than decomposability
Structured decomposability enables tractable ...

- **Entropy** of probabilistic circuit [Liang et al. 2017]
- **Symmetric** and **group queries** (exactly-$k$, odd-number, more, etc.) [Bekker et al. 2015]

For the “right” vtree

- Probability of logical circuit event in probabilistic circuit [ibid.]
- **Multiply** two probabilistic circuits [Shen et al. 2016]
- **KL Divergence** between probabilistic circuits [Liang et al. 2017]
- **Same-decision probability** [Oztok et al. 2016]
- **Expected same-decision probability** [Choi et al. 2017]
- **Expected classifier agreement** [Choi et al. 2018]
- **Expected predictions** [Khosravi et al. 2019b]
Structured decomposability enables tractable ...

- **Entropy** of probabilistic circuit [Liang et al. 2017]
- **Symmetric** and **group queries** (exactly-$k$, odd-number, more, etc.) [Bekker et al. 2015]

For the “right” vtree

- Probability of logical circuit event in probabilistic circuit [ibid.]
- **Multiply** two probabilistic circuits [Shen et al. 2016]
- **KL Divergence** between probabilistic circuits [Liang et al. 2017]
- **Same-decision probability** [Oztok et al. 2016]
- **Expected same-decision probability** [Choi et al. 2017]
- **Expected classifier agreement** [Choi et al. 2018]
- **Expected predictions** [Khosravi et al. 2019b]
Stay Tuned For ...

Next:
1. How probabilistic circuits are related to logical ones?
   ⇒ a historical perspective
2. How probabilistic circuits in the literature relate and differ?
   ⇒ SPNs, ACs, CNets, PSDDs
3. How classical tractable models can be turned in a circuit?
   ⇒ Compiling low-treewidth PGMs

After: How do I build my own probabilistic circuit?
Tractability to other semi-rings

Tractable probabilistic inference exploits **efficient summation for decomposable functions** in the probability commutative semiring:

$$(\mathbb{R}, +, \times, 0, 1)$$

analogously efficient computations can be done in other semi-rings:

$$(\mathcal{S}, \oplus, \otimes, 0_{\oplus}, 1_{\otimes})$$

⇒ **Algebraic model counting** [Kimmig et al. 2017], **Semi-ring programming** [Belle et al. 2016]

Historically, **very well studied for boolean functions**:

$$(\mathbb{B} = \{0, 1\}, \lor, \land, 0, 1)$$

⇒ **logical circuits!**
Logical circuits

Logical circuits are compact representations for boolean functions...

s/d-D/DNFs
[Darwiche et al. 2002]

O/BDDs
[Bryant 1986]

SDDs
[Darwiche 2011]
Logical circuits

structural properties

...and as probabilistic circuits, one can define **structural properties**: (structured) decomposability, smoothness, determinism allowing for tractable computations

---

Darwiche et al., “A knowledge compilation map”, 2002
Logical circuits

a knowledge compilation map

...inducing a hierarchy of tractable query classes

Darwiche et al., “A knowledge compilation map”, 2002
A task called **weighted model counting** (WMC)

\[
WMC(\Delta, w) = \sum_{x \models \Delta} \prod_{l \in x} w(l)
\]

Two decades worth of connections:

1. Encode probabilistic model as WMC (add variable placeholders for parameters)
2. Compile $\Delta$ into a d-DNNF (or OBDD, SDD, etc.)
3. Tractable MAR/CON by tractable WMC on circuit
4. Depending on the WMC encoding even tractable MAP

End result equivalent to probabilistic circuit: efficiently replace parameter variables in logical circuit by edge parameters in probabilistic circuit
From trees to circuits
via compilation
From trees to circuits
via compilation

Bottom-up *compilation*: starting from leaves...
...compile a leaf CPT

\[ p(A|C = 0) \]

\[ A = 0 \quad A = 1 \]
From trees to circuits

via compilation

...compile a leaf CPT

\[ p(A|C = 1) \]

\[ \begin{align*}
A = 0 & \quad 0.6 \\
A = 1 & \quad 0.4
\end{align*} \]
From trees to circuits
via compilation

...compile a leaf CPT...for all leaves...

\[ p(A|C) \]
\[ p(B|C) \]

\[ \begin{array}{ccc}
A = 0 & A = 1 & B = 0 \\
\hline
B = 0 & B = 1 & \\
\end{array} \]
From trees to circuits
via compilation

...and recurse over parents...

\( p(C|D = 0) \)
From trees to circuits
via compilation

...while reusing previously compiled nodes!...

\[ p(C|D = 1) \]

\[ \begin{array}{c c c c}
  A = 0 & A = 1 & B = 0 & B = 1 \\
  C = 0 & + & + & + \\
  C = 1 & \times & \times & \times \\
\end{array} \]

\[ \begin{array}{c}
  p(C|D = 1) \end{array} \]
From trees to circuits
via compilation

\[ A = 0 \quad A = 1 \quad B = 0 \quad B = 1 \]
\[ C = 0 \quad C = 1 \]
\[ D = 0 \quad D = 1 \]

\[ p(D) \]

\[ \times \times \times \times \]

\[ \times \times \times \times \]
**Low-treewidth PGMs**

Tree, polytrees and thin junction trees can be turned into:
- decomposable
- smooth
- deterministic

Therefore they support tractable:
- EVI
- MAR/CON
- MAP
Arithmetic Circuits (ACs)

ACs [Darwiche 2003] are
- decomposable
- smooth
- deterministic

They support tractable
- EVI
- MAR/CON
- MAP

⇒ parameters are attached to the leaves
⇒ ...but can be moved to the sum node edges
⇒ Also see related AND/OR search spaces [Dechter et al. 2007]

Lowd et al., “Learning Markov Networks With Arithmetic Circuits”, 2013
Sum-Product Networks (SPNs)

SPNs [Poon et al. 2011] are
- decomposable
- smooth
- deterministic

They support tractable
- EVI
- MAR/CON
- MAP

⇒ deterministic SPNs are also called selective [Peharz et al. 2014]
A CNet [Rahman et al. 2014] is a **weighted model-trees** [Dechter et al. 2007] whose leaves are tree Bayesian networks.

⇒ they can be represented as probabilistic circuits
**CNets as probabilistic circuits**

Every *decision node* in the CNet can be represented as a deterministic, smooth sum node

and we can recurse on each child node until a BN tree is reached

⇒ *compilable into a deterministic, smooth and decomposable circuit!*
CNets as probabilistic circuits

CNets are
- decomposable
- smooth
- deterministic

They support tractable
- EVI
- MAR/CON
- MAP

⇒ EVI can be computed in $O(|X|)$
Probabilistic Sentential Decision Diagrams

PSDDs [Kisa et al. 2014a] are
- structured
- decomposable
- smooth
- deterministic

They support tractable
- EVI
- MAR/CON
- MAP
- Complex queries!

Kisa et al., “Probabilistic sentential decision diagrams”, 2014
Shen et al., “Conditional PSDDs: Modeling and learning with modular knowledge”, 2018
where are probabilistic circuits?
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., “Probabilistic deep learning using random sum-product networks”, 2018
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
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
Building circuits
Read more in online slides about ...

Building Circuits:

1. How to learn circuit parameters?
   ⇒ convex optimization, EM, SGD, Bayesian learning, ...

2. How to learn the structure of circuits?
   ⇒ local search, random structures, ensembles, ...

3. How to compile other models to circuits?
   ⇒ PGM compilation, probabilistic databases, probabilistic programming

A learner $L$ is a tractable learner for a class of queries $Q$ iff
(1) for any dataset $D$, learner $L(D)$ runs in time $O(poly(|D|))$, and
(2) outputs a probabilistic model that is tractable for queries $Q$. 
Tractable Learning

A learner $L$ is a tractable learner for a class of queries $Q$ iff

1. for any dataset $D$, learner $L(D)$ runs in time $O(poly(|D|))$, and

   $\Rightarrow$ Guarantees learned model has size $O(poly(|D|))$

   $\Rightarrow$ Guarantees learned model has size $O(poly(|X|))$

2. outputs a probabilistic model that is tractable for queries $Q$. 
Tractable Learning

A learner $L$ is a tractable learner for a class of queries $Q$ iff

1. for any dataset $D$, learner $L(D)$ runs in time $O(poly(|D|))$, and

   $\Rightarrow$ Guarantees learned model has size $O(poly(|D|))$

   $\Rightarrow$ Guarantees learned model has size $O(poly(|X|))$

2. outputs a probabilistic model that is tractable for queries $Q$.

   $\Rightarrow$ Guarantees efficient querying for $Q$ in time $O(poly(|X|))$
Applications
Applications:

1. what have been probabilistic circuits used for?
   \[\Rightarrow\] computer vision, sop, speech, planning, ...

2. what are the current trends in tractable learning?
   \[\Rightarrow\] hybrid models, probabilistic programming, ...

3. what are the current challenges?
   \[\Rightarrow\] benchmarks, scaling, reasoning

Chavira et al., “Compiling relational Bayesian networks for exact inference”, 2006
Holtzen et al., “Symbolic Exact Inference for Discrete Probabilistic Programs”, 2019
Vlasselaer et al., “Exploiting Local and Repeated Structure in Dynamic Bayesian Networks”, 2016
takeaway #1 tractability is a spectrum
takeaway #2: you can be both tractable and expressive
takeaway #3: **probabilistic circuits are a foundation for tractable inference and learning**
Tractable Probabilistic Circuits @ ICLP?

- Logical roots of probabilistic circuits
- Probabilistic circuits bridge between logic and deep learning
- Bring back world models!
- Powerful general reasoning tool
  - for example in probabilistic logic programming
- Elegant knowledge representation formalism
References


References II


Darwiche, Adnan (2003). “A Differential Approach to Inference in Bayesian Networks”. In: JACM.


References III


References IV


References V


References VI


References VII


Khosravi, Pasha et al. (2019b). “What to Expect of Classifiers? Reasoning about Logistic Regression with Missing Features”. In: *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI)*.
References VIII
