Probabilistic Generating Circuits - Demystified

Sanyam Agarwal and Markus Bläser

Saarland University

Probabilistic Inference

Probabilistic models should:

- Allow for efficient probabilistic inference.
- Represent as many classes of distributions as possible

Probabilistic Inference

Probabilistic models should:

- Allow for efficient probabilistic inference.
- Represent as many classes of distributions as possible

But n binary random $\implies 2^n$ assignments.

Probabilistic Inference

Probabilistic models should:

- Allow for efficient probabilistic inference.
- Represent as many classes of distributions as possible

But n binary random $\implies$ 2^n assignments.

Tradeoff between expressiveness and tractability. [Rot96].

Examples of tractable models

- Determinantal Point Processes (DPPs)
  - Characterized by a PSD matrix.
  - Model negative dependencies.

Examples of tractable models

- Determinantal Point Processes (DPPs)
  - Characterized by a PSD matrix.
  - Model negative dependencies.
- Probabilistic Circuits (PCs)
  - Store probability mass functions.
  - **Only** allow positive weights.

Examples of tractable models

- Determinantal Point Processes (DPPs)
  - Characterized by a PSD matrix.
  - Model negative dependencies.
- Probabilistic Circuits (PCs)
  - Store probability mass functions.
  - **Only** allow positive weights.
  - * Power is incomparable to DPPs. [ZHV20]

Examples of tractable models

- Determinantal Point Processes (DPPs)
  - Characterized by a PSD matrix.
  - Model negative dependencies.
- Probabilistic Circuits (PCs)
  - Store probability mass functions.
  - **Only** allow positive weights.
  - * Power is incomparable to DPPs. [ZHV20]
- Probabilistic Generating Circuits (PGCs)
  - Store probability generating polynomial.
  - Allow for negative weights.

Examples of tractable models

- Determinantal Point Processes (DPPs)
  - Characterized by a PSD matrix.
  - Model negative dependencies.
- Probabilistic Circuits (PCs)
  - Store probability mass functions.
  - **Only** allow positive weights.
  - * Power is incomparable to DPPs. [ZHV20]
- Probabilistic Generating Circuits (PGCs)
  - Store probability generating polynomial.
  - Allow for negative weights.
  - * PGCs tractably subsume both PCs and DPPs. [ZJV21]

Examples of tractable models

- Determinantal Point Processes (DPPs)
  - Characterized by a PSD matrix.
  - Model negative dependencies.
- Probabilistic Circuits (PCs)
  - Store probability mass functions.
  - **Only** allow positive weights.
  - * Power is incomparable to DPPs. [ZHV20]
- Probabilistic Generating Circuits (PGCs)
  - Store probability generating polynomial.
  - Allow for negative weights.
  - * PGCs tractably subsume both PCs and DPPs. [ZJV21]

Motivation for our work

- Where does this power of PGCs comes from?

Examples of tractable models

- Determinantal Point Processes (DPPs)
  - Characterized by a PSD matrix.
  - Model negative dependencies.
- Probabilistic Circuits (PCs)
  - Store probability mass functions.
  - **Only** allow positive weights.
  - * Power is incomparable to DPPs. [ZHV20]
- Probabilistic Generating Circuits (PGCs)
  - Store probability generating polynomial.
  - Allow for negative weights.
  - * PGCs tractably subsume both PCs and DPPs. [ZJV21]

Motivation for our work

- Where does this power of PGCs comes from?
- How powerful they truly are?

Source of power of PGCs*

The power of PGCs comes from negative weights, as PCs with negative weights (*nonmonotone PCs*) subsume PGCs.

Source of power of PGCs*

The power of PGCs comes from negative weights, as PCs with negative weights (*nonmonotone PCs*) subsume PGCs.

Extent of power of PGCs*

Efficient marginalization over PGCs representing distributions with more than two categories implies P = NP.

Source of power of PGCs*

The power of PGCs comes from negative weights, as PCs with negative weights (*nonmonotone PCs*) subsume PGCs.

Extent of power of PGCs*

Efficient marginalization over PGCs representing distributions with more than two categories implies P = NP.

Power of nonmonotone PCs

Nonmonotone PCs computing set-multilinear polynomials support tractable marginalization over categorical variables of an arbitrary image size.

*Proven independently in [BZV24].

DPP example

$$L = \begin{array}{cccc} X_1 & X_2 & X_3 & \\ 4 & 1 & 3 & X_1 \\ 1 & 5 & 2 & X_2 \\ 3 & 2 & 8 & X_3 \end{array}$$

$$Pr(X_1 = 0, X_2 = 1, X_3 = 1)$$

DPP example

$$L = \begin{array}{cccc} X_1 & X_2 & X_3 & \\ 4 & 1 & 3 & X_1 \\ 1 & 5 & 2 & X_2 \\ 3 & 2 & 8 & X_3 \end{array}$$

$$Pr(X_1 = 0, X_2 = 1, X_3 = 1)$$

$$\propto \begin{vmatrix} 5 & 2 \\ 2 & 8 \end{vmatrix}$$

Normalization Constant $= det(L + I) = 199$

DPP example

$$L = \begin{array}{cccc} X_1 & X_2 & X_3 & \\ 4 & 1 & 3 & X_1 \\ 1 & 5 & 2 & X_2 \\ 3 & 2 & 8 & X_3 \end{array}$$

$$Pr(X_1 = 0, X_2 = 1, X_3 = 1)$$

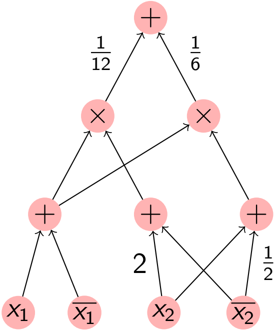
$$\propto \begin{vmatrix} 5 & 2 \\ 2 & 8 \end{vmatrix}$$

Normalization Constant $= det(L + I) = 199$

Allow tractable marginalization by storing probabilities as determinants of submatrices.

PC and PGC example

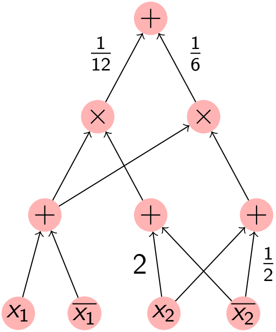
Pr	$X_1 = 0$	$X_1 = 1$
$X_2 = 0$	$\frac{1}{6}$	$\frac{1}{6}$
$X_2 = 1$	$\frac{1}{3}$	$\frac{1}{3}$



$$f = \frac{1}{6}\overline{x_1}\,\overline{x_2} + \frac{1}{3}\overline{x_1}x_2 + \frac{1}{6}x_1\overline{x_2} + \frac{1}{3}x_1x_2$$

PC and PGC example

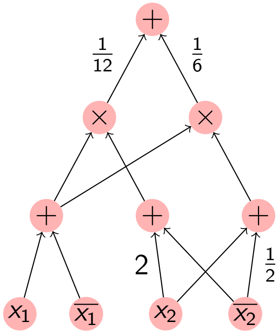
Pr	$X_1 = 0$	$X_1 = 1$
$X_2 = 0$	$\frac{1}{6}$	$\frac{1}{6}$
$X_2 = 1$	$\frac{1}{3}$	$\frac{1}{3}$

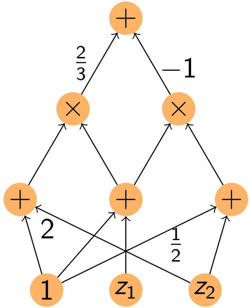


Marginalisation is tractable **if** PC is decomposable and smooth.

PC and PGC example

Pr	$X_1 = 0$	$X_1 = 1$
$X_2 = 0$	$\frac{1}{6}$	$\frac{1}{6}$
$X_2 = 1$	$\frac{1}{3}$	$\frac{1}{3}$



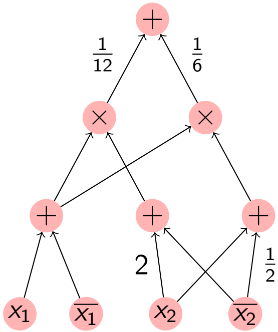


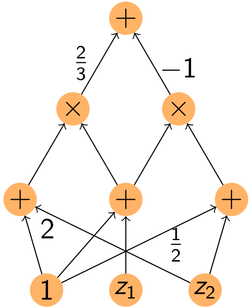
Marginalisation is tractable **if** PC is decomposable and smooth.

$$f = \frac{1}{6} + \frac{1}{3}z_2 + \frac{1}{6}z_1 + \frac{1}{3}z_1 z_2$$

PC and PGC example

Pr	$X_1 = 0$	$X_1 = 1$
$X_2 = 0$	$\frac{1}{6}$	$\frac{1}{6}$
$X_2 = 1$	$\frac{1}{3}$	$\frac{1}{3}$





Marginalisation is tractable **if** PC is decomposable and smooth.

Tractable marginalization for binary variables.

Separation between PCs and PGCs

Theorem

PGCs over binary variables can be simulated by nonmonotone PCs with only polynomial overhead in size.

Separation between PCs and PGCs

Theorem

PGCs over binary variables can be simulated by nonmonotone PCs with only polynomial overhead in size.

Proof Idea.

- Let a PGC compute $f(z_1, ..., z_n)$. Define the polynomial g as

$$g(x_1, \overline{x_1}, ..., x_n, \overline{x_n}) = f(\frac{x_1}{\overline{x_1}}, \frac{x_2}{\overline{x_2}}, ..., \frac{x_n}{\overline{x_n}}) \cdot \prod_{i=1}^{n} \overline{x_i}.$$

Separation between PCs and PGCs

Theorem

PGCs over binary variables can be simulated by nonmonotone PCs with only polynomial overhead in size.

Proof Idea.

- Let a PGC compute $f(z_1, ..., z_n)$. Define the polynomial g as

$$g(x_1, \overline{x_1}, ..., x_n, \overline{x_n}) = f(\frac{x_1}{\overline{x_1}}, \frac{x_2}{\overline{x_2}}, ..., \frac{x_n}{\overline{x_n}}) \cdot \prod_{i=1}^{n} \overline{x_i}.$$

- g would be *multilinear* and *homogenous* with $deg(g) = n$
- Multiplicative term only increases size of g by $O(n)$.

Separation between PCs and PGCs

Theorem

PGCs over binary variables can be simulated by nonmonotone PCs with only polynomial overhead in size.

Proof Idea.

- Let a PGC compute $f(z_1, ..., z_n)$. Define the polynomial g as

$$g(x_1, \overline{x_1}, ..., x_n, \overline{x_n}) = f(\frac{x_1}{\overline{x_1}}, \frac{x_2}{\overline{x_2}}, ..., \frac{x_n}{\overline{x_n}}) \cdot \prod_{i=1}^{n} \overline{x_i}.$$

- g would be *multilinear* and *homogenous* with $deg(g) = n$
- Multiplicative term only increases size of g by $O(n)$.
- To remove division gates, use [Str73]'s result to eliminate them in polynomial overhead.

Non-binary marginalization is hard for PGCs

Theorem

Efficient marginalization over PGCs involving quaternary random variables implies that NP = P.

Non-binary marginalization is hard for PGCs

Theorem

Efficient marginalization over PGCs involving quaternary random variables implies that NP $=$ P.

Proof Idea.

- $\#PM$ in 3-regular bipartite graph G is $\#$P-Hard. [DL92]

Non-binary marginalization is hard for PGCs

Theorem

Efficient marginalization over PGCs involving quaternary random variables implies that NP = P.

Proof Idea.

- $\#PM$ in 3-regular bipartite graph G is #P-Hard. [DL92]
- Using G, we construct a PGC $C = \prod_{i=1}^{n} \sum_{j \in N(i)} E_{i,j} V_j$.
- Each V_i appears thrice, hence acts as quaternary variable.

Non-binary marginalization is hard for PGCs

Theorem

Efficient marginalization over PGCs involving quaternary random variables implies that NP $=$ P.

Proof Idea.

- $\#PM$ in 3-regular bipartite graph G is #P-Hard. [DL92]
- Using G, we construct a PGC $C = \prod_{i=1}^{n} \sum_{j \in N(i)} E_{i,j} V_j$.
- Each V_i appears thrice, hence acts as quaternary variable.
- Number of monomials in coefficient of $V_1 \cdot V_2 \cdot \ldots \cdot V_n$ gives us number of perfect matchings in G.

□

Non-binary marginalization is hard for PGCs

Theorem

Efficient marginalization over PGCs involving quaternary random variables implies that NP $=$ P.

Proof Idea.

- $\#PM$ in 3-regular bipartite graph G is #P-Hard. [DL92]
- Using G, we construct a PGC $C = \prod_{i=1}^{n} \sum_{j \in N(i)} E_{i,j} V_j$.
- Each V_i appears thrice, hence acts as quaternary variable.
- Number of monomials in coefficient of $V_1 \cdot V_2 \cdot \ldots \cdot V_n$ gives us number of perfect matchings in G.

□

Remark: Similar result follows for ternary variables, if we allow marginalization over subsets of images.

Tractable marginalization in nonmonotone PCs

Theorem

Let C be a nonmonotone PC of size s computing a probability distribution over categorical random variables $X_1, \ldots, X_n$ such that the polynomial P computed by C is *set-multilinear*. Then we can marginalize in linear time.

Tractable marginalization in nonmonotone PCs

Theorem

Let C be a nonmonotone PC of size s computing a probability distribution over categorical random variables $X_1, \ldots, X_n$ such that the polynomial P computed by C is *set-multilinear*. Then we can marginalize in linear time.

- **Definition (Set-multilinear)**: Let $V = Y_1 \sqcup Y_2 \sqcup ... \sqcup Y_k$. A polynomial p on a set of variables V is set multilinear, **if** every monomial of p contains exactly one variable from each Y_i for all i.

Tractable marginalization in nonmonotone PCs

Theorem

Let C be a nonmonotone PC of size s computing a probability distribution over categorical random variables $X_1, \ldots, X_n$ such that the polynomial P computed by C is *set-multilinear*. Then we can marginalize in linear time.

- **Definition (Set-multilinear)**: Let $V = Y_1 \sqcup Y_2 \sqcup \ldots \sqcup Y_k$. A polynomial p on a set of variables V is set multilinear, **if** every monomial of p contains exactly one variable from each Y_i for all i.
- Each X_i represented by partition $z_{i,0}, \ldots, z_{i,d-1}$ which are inputs to C.

Connection with DPPs

- While PGCs subsume DPPs, a related question is:
  What happens to the power of DPPs under simple pre- and post-processing?

Connection with DPPs

- While PGCs subsume DPPs, a related question is:
  What happens to the power of DPPs under simple pre- and post-processing?
- We show that this question is **Hard** to answer.

Theorem

Any arithmetic formula can be represented as an affine projection of a *DPP*.

Interpretation: If there is a PGC that cannot be written as affine projection of a DPP, then we separate algebraic formulas and circuits.

Conclusions and Future Work

Summary

- Power of PGCs comes from negative weights.

Conclusions and Future Work

Summary

- Power of PGCs comes from negative weights.
- PGCs allow efficient marginalisation **only** for binary variables.

Conclusions and Future Work

Summary

- Power of PGCs comes from negative weights.
- PGCs allow efficient marginalisation **only** for binary variables.
- Difficult to establish connection between PGCs and DPPs under pre and post-processing.

Conclusions and Future Work

Summary

- Power of PGCs comes from negative weights.
- PGCs allow efficient marginalisation **only** for binary variables.
- Difficult to establish connection between PGCs and DPPs under pre and post-processing.

Future Directions

- Are there efficient marginalisation algorithms for categorical distributions which are not set-multilinear?

Conclusions and Future Work

Summary

- Power of PGCs comes from negative weights.
- PGCs allow efficient marginalisation **only** for binary variables.
- Difficult to establish connection between PGCs and DPPs under pre and post-processing.

Future Directions

- Are there efficient marginalisation algorithms for categorical distributions which are not set-multilinear?
- What is the most general class of models that represents tractable distributions?

References

[BZV24] O. Broadrick, H. Zhang, and G. Van den Broeck. "Polynomial Semantics of Tractable Probabilistic Circuits". In: *UAI* (2024).

[DL92] P. Dagum and M. Luby. "Approximating the permanent of graphs with large factors". In: *TCS* (1992).

[Rot96] D. Roth. "On the hardness of approximate reasoning". In: *AI* (1996).

[Str73] V. Strassen. "Vermeidung von Divisionen". In: *Journal für die reine und angewandte Mathematik* (1973).

[ZHV20] H. Zhang, S. Holtzen, and G. Van den Broeck. "On the Relationship Between Probabilistic Circuits and Determinantal Point Processes". In: *UAI*. 2020.

[ZJV21] H. Zhang, B. Juba, and G. Van Den Broeck. "Probabilistic Generating Circuits". In: *ICML*. 2021.

Thank You!

Questions?