On Inference and Learning With Probabilistic Generating Circuits

Juha Harviainen

UNIVERSITY OF HELSINKI

Vaidyanathan Peruvemba Ramaswamy

TU WIEN

Mikko Koivisto

UNIVERSITY OF HELSINKI

UAI 2023

Overview

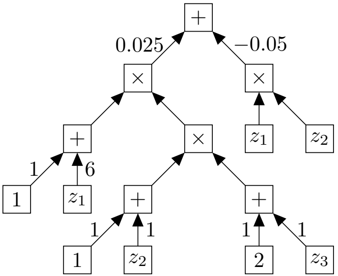
Introduction

Probabilistic Generating Circuit (PGC)

- **Tractable probabilistic model** (TPM)
- Encodes the joint probability distribution of binary variables
- **Probability generating polynomial** (PGP)

$$0.150 z_1 z_2 z_3 + 0.025 z_2 z_3 + 0.150 z_1 z_3 + 0.025 z_3$$
$$+ 0.250 z_1 z_2 + 0.050 z_2 + 0.300 z_1 + 0.050$$

X_1	X_2	X_3	$\Pr(\cdot)$
0	0	0	0.050
1	0	0	0.300
0	1	0	0.050
1	1	0	0.250
0	0	1	0.025
1	0	1	0.150
0	1	1	0.025
1	1	1	0.150

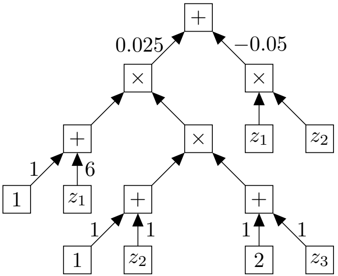


Probabilistic Generating Circuit (PGC)

- **Tractable probabilistic model** (TPM)
- Encodes the joint probability distribution of binary variables
- **Probability generating polynomial** (PGP)

$$0.150z_1z_2z_3 + 0.025z_2z_3 + 0.150z_1z_3 + 0.025z_3$$
$$+ 0.250z_1z_2 + 0.050z_2 + 0.300z_1 + 0.050$$

X_1	X_2	X_3	$\Pr(\cdot)$
0	0	0	0.050
1	0	0	0.300
0	1	0	0.050
1	1	0	0.250
0	0	1	0.025
1	0	1	0.150
0	1	1	0.025
1	1	1	0.150

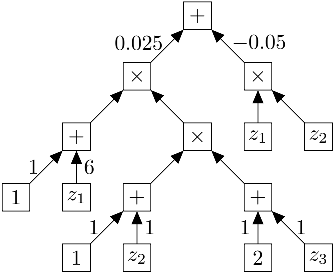


Probabilistic Generating Circuits

- Polynomial-time marginal inference[1]
- **Goal**: Evaluate $\Pr(\{X_i = 1\}_{i\in A}, \{X_j = 1\}_{j \in B})$
- **Solution**: Assign

$$z_i \mapsto \begin{cases} t, & i \in A \\ 0, & i \in B \\ 1, & i \notin A \cup B \end{cases}$$

- Extract the coefficient of the term $t^{|A|}$



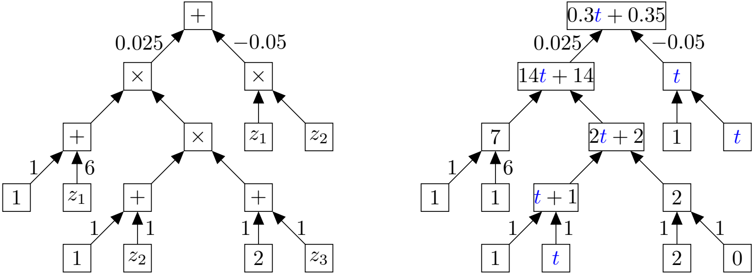
[1] Zhang, H., Juba, B., Van den Broeck, G. Probabilistic Generating Circuits. *ICML'21*.

Probabilistic Generating Circuits

- **Solution**: Assign

$$z_i \mapsto \begin{cases} t, & i \in A \\ 0, & i \in B \\ 1, & i \notin A \cup B \end{cases}$$

- Extract the coefficient of the term $t^{|A|}$
- **Example**: Let $A = \{2\}$ and $B = \{3\}$
- We have $\Pr(X_2 = 1, X_3 = 0) = 0.3$

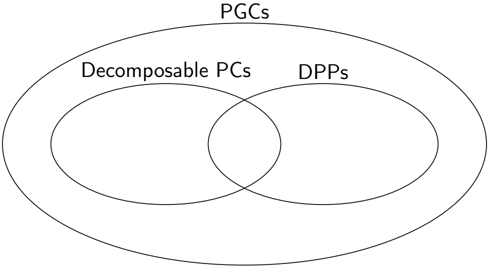


Relationship to Other TPMs

PGCs strictly more expressive efficient than[1]

- Decomposable **probabilistic circuits** (PCs)
- **Determinantal point processes** (DPPs)

Above TPMs' relationship studied by Zhang et al.[2]



[2] Zhang, H., Holtzen, S., Van den Broeck, G. On the Relationship Between Probabilistic Circuits and Determinantal Point Processes. *UAI'20*.

Our Contributions

Speedups on marginal probability queries **in three different cases**. *[this talk]*

A proof of **NP-completeness** of detecting **infeasible PGCs**.

A sketch of a framework for **learning PGCs** that avoids this issue. *[this talk]*

Faster Inference

General Case

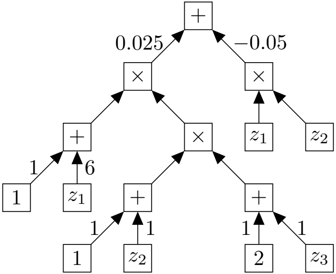
- **Previous work**: $O(mn \log n \log \log n)$ with m nodes and n variables
- At most m operations over polynomials of degree n
- Consider instead assigning $t = 0, 1, \ldots, |A|$ in time $O(mn)$
- This determines the univariate polynomial uniquely
- Apply polynomial interpolation and extract the coefficient of $t^{|A|}$

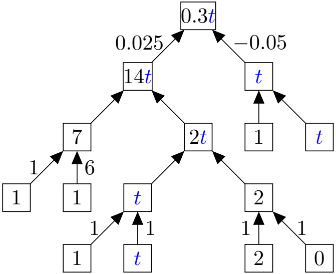
Decomposable Circuits

- **Decomposability**: no indeterminate appears in both subcircuits of any multiplication gate
- For decomposable PGCs, it is sufficient to track highest–degree terms

Theorem

There is a ***linear-time algorithm*** *for marginal inference on decomposable PGCs*





Determinantal Forms

- A **determinant** of a matrix can encode a PGP:

$$\frac{1}{13}\det\begin{pmatrix} z_1+2z_2 & z_1 & z_1 & -z_1 \\ 1+z_2 & -z_3 & z_5 & 0 \\ z_2 & 1 & 2 & z_4 \\ -z_2 & 0 & z_5 & 0 \end{pmatrix}$$

$$= \frac{1}{13}(2z_1z_2z_3 + z_1z_2z_3z_4 + z_1z_2z_3z_5 + \cdots + 2z_2z_3z_4z_5)$$

- Support for marginal queries by evaluating a **real-valued** determinant
- With $A = \{1,2,3\}$ and $B = \{4\}$ we want terms $2z_1z_2z_3$ and $z_1z_2z_3z_5$
- Assign values for indeterminates and zero out certain entries

$$\frac{1}{13}\det\begin{pmatrix} 0 & 0 & 1 & -1 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & 2 & 0 \\ -1 & 0 & 1 & 0 \end{pmatrix} = \frac{1}{13}(2+1)$$

Learning

Learning is Hard

- Unfortunately, learning PGCs seems to be hard:

Theorem

*It is **NP-hard** to check if a PGC encodes a valid probability distribution*

- SAT is reducible to finding a term with a negative coefficient
- **Moderate-size circuits still verifiable!**

Learning Compound Circuits

- Have the PGC consist of many subcircuits
- Subcircuits verifiable independently

Markov Chain simulation

1. *Initialize:* Pick any feasible solution
2. *Move:* Pick a candidate solution from the local neighborhood
3. *Accept or reject:* Replace current solution by the candidate if valid
4. *Iterate:* Go to step 2

- Each step an engineering problem of its own
- Implementation left for future work

Concluding Remarks

Concluding Remarks

- Inference is fast but learning is hard
- Many open implementation details
- Are there classes of PGCs that are fast to validate (on average)

Thank you!