Lower Bounds for Tractable Arithmetic Circuits

Alexis de Colnet

ALGORITHMS AND COMPLEXITY GROUP

TU WIEN

Roadmap

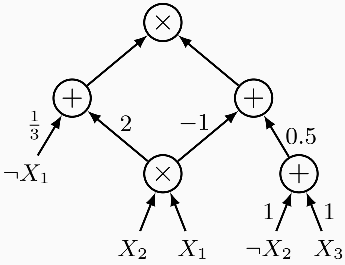
The talk is **not** intended to be an exhaustive list of all lower bounds techniques for ACs. I will mostly cover results from [dC and Mengel, 2021].

- Tractable arithmetic circuits (AC)
- Reduction to tractable Boolean circuits (the boring trick)
- Lower bounds via the rank technique
- Separating classes of tractable ACs

Arithmetic Circuits

An arithmetic circuit (AC) is a computational directed acyclic graph s.t.

- it has a single sink (the output gate)
- its sources correspond to Boolean 0/1 literals (the input gates)
- its internal nodes correspond to $\times$ or $+$ operations ($\times$-gates and $+$-gates)
- the input connectors of its $+$-gates are weighted by rational numbers



These circuits have been given different names: $(+, \times)$-programs [Valiant, 1980], AC [Nisan and Wigderson, 1997; Darwiche 2002], sum-product networks [Poon and Domingos, 2011; Dennis 2016], etc.

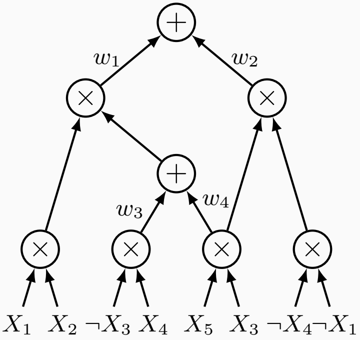
Arithmetic Circuits

In this talk, all ACs represent non-negative functions. Though we allow negative weights on the edges.

An AC is called monotone if it only uses non-negative weights.

It is known that allowing negative weights ($\simeq$ allowing subtraction) can result in a more-than-polynomial decrease in the size of the AC [Valiant, 1980]. So we aim for lower bounds on the size of AC representing non-negative functions but where negative weights are allowed.

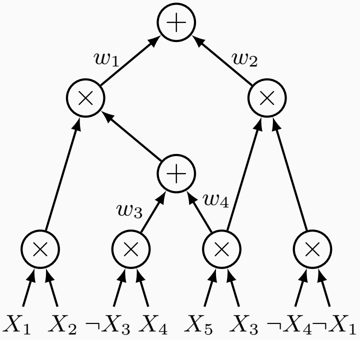
- **Decomposability**: every $\times$-gate $C_1 \times C_2$ verifies $var(C_1) \cap var(C_2) = \emptyset$ (also called syntactical multilinearity)



Tractable Arithmetic Circuits

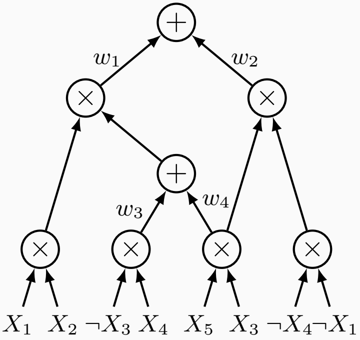
- **Decomposability**: every $\times$-gate $C_1 \times C_2$ verifies $var(C_1) \cap var(C_2) = \emptyset$ (also called syntactical multilinearity)

- **Determinism**: every $+$-gate $w_1 C_1 + w_2 C_2$ verifies that $\forall \vec{X}, C_1(\vec{X}) \cdot C_2(\vec{X}) = 0$ (i.e., C_1 and C_2 have disjoint supports)



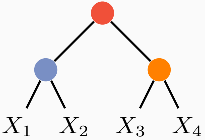
Tractable Arithmetic Circuits

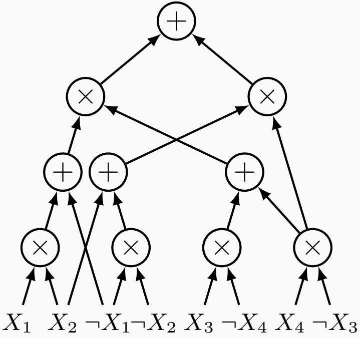
- **Decomposability**: every $\times$-gate $C_1 \times C_2$ verifies $var(C_1) \cap var(C_2) = \emptyset$ (also called syntactical multilinearity)
- **Determinism**: every $+$-gate $w_1 C_1 + w_2 C_2$ verifies that $\forall \vec{X}, C_1(\vec{X}) \cdot C_2(\vec{X}) = 0$ (i.e., C_1 and C_2 have disjoint supports)
- **Smoothness**: every $+$-gate $C_1 + C_2$ verifies $var(C_1) = var(C_2)$



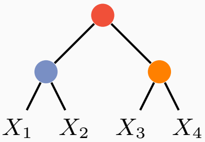
Tractable Arithmetic Circuits

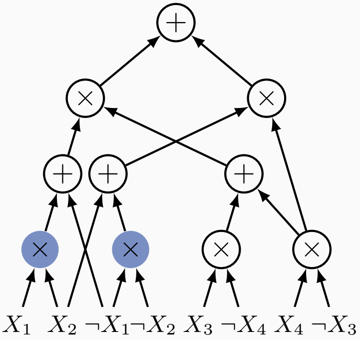
- **Structured-decomposability**: there is a **vtree** (variable tree) T such that for every $\times$-gate $C_1 \times C_2$, there is a node $t \in T$ that **separates** $var(C_1)$ from $var(C_2)$. I.e., t's children t_1, t_2 are such that $var(C_1) \subseteq var(t_1)$ and $var(C_2) \subseteq var(t_2)$



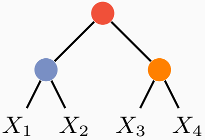


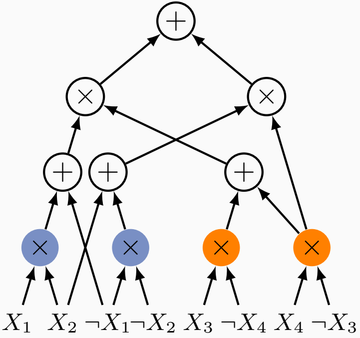
- **Structured-decomposability**:
  there is a **vtree** (variable tree) T such that for every $\times$-gate $C_1 \times C_2$, there is a node $t \in T$ that **separates** $var(C_1)$ from $var(C_2)$. I.e., t's children t_1, t_2 are such that $var(C_1) \subseteq var(t_1)$ and $var(C_2) \subseteq var(t_2)$



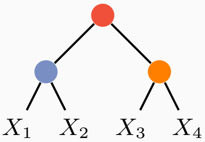


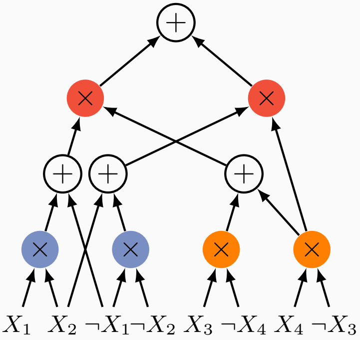
- **Structured-decomposability**: there is a **vtree** (variable tree) T such that for every $\times$-gate $C_1 \times C_2$, there is a node $t \in T$ that **separates** $var(C_1)$ from $var(C_2)$. I.e., t's children t_1, t_2 are such that $var(C_1) \subseteq var(t_1)$ and $var(C_2) \subseteq var(t_2)$



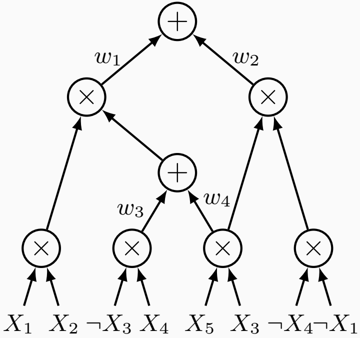


- **Structured-decomposability**: there is a **vtree** (variable tree) T such that for every $\times$-gate $C_1 \times C_2$, there is a node $t \in T$ that **separates** $var(C_1)$ from $var(C_2)$. I.e., t's children t_1, t_2 are such that $var(C_1) \subseteq var(t_1)$ and $var(C_2) \subseteq var(t_2)$



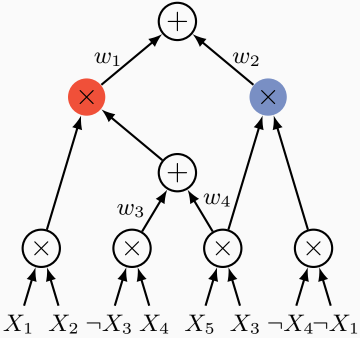


Tractable Arithmetic Circuits



This AC is decomposable but not structured-decomposable.

Tractable Arithmetic Circuits



This AC is decomposable but not structured-decomposable. If it was structured by a vtree T:

- there would be a node in T that separates $\{X_1\}$ from $\{X_3, X_4\}$,
- and there would be a node in T that separates $\{X_1, X_4\}$ from $\{X_3\}$,

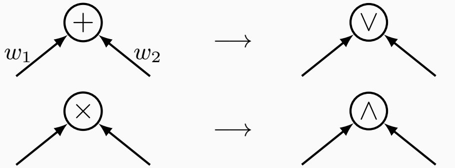
this is not possible.

Roadmap

- ~~Tractable arithmetic circuits (AC)~~
- Reduction to tractable Boolean circuits (the boring trick)
- Lower bounds via the rank technique
- Separating classes of tractable ACs

Reduction to Tractable Boolean Circuits – the Boring Trick

There is a trivial poly-time transformation from a tractable AC that uses only positive weights to a tractable Boolean circuit.



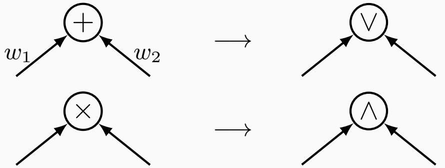
$$C(\boldsymbol{X}) \longrightarrow \mathbb{1}\left[C(\boldsymbol{X}) > 0\right]$$

decomposable AC	$\longrightarrow$	DNNF circuits
det. and dec. AC	$\longrightarrow$	d-DNNF circuits
structured-dec. AC	$\longrightarrow$	SDNNF circuits

Thus, if the support of a function f coincides with the true points of a Boolean function that admits only exponential size DNNF circuits, then f admits only exponential size decomposable AC that use only positive weights.

Reduction to Tractable Boolean Circuits – the Boring Trick

There is a trivial poly-time transformation from a tractable AC that uses only positive weights to a tractable Boolean circuit.



$$C(\boldsymbol{X}) \longrightarrow \mathbb{1}\left[C(\boldsymbol{X}) > 0\right]$$

decomposable AC $\longrightarrow$ DNNF circuits
det. and dec. AC $\longrightarrow$ d-DNNF circuits
structured-dec. AC $\longrightarrow$ SDNNF circuits

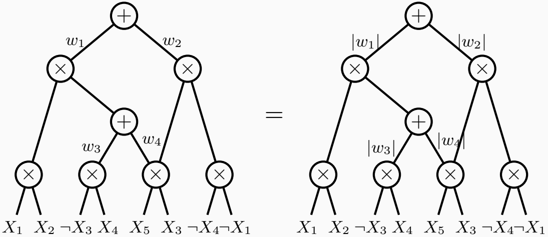
Tractable Boolean circuits. We know many hard functions. [Beame, Li, Roy and Suciu, 2013; Bova, Capelli, Mengel and Slivovsky, 2014 and 2016; Amarilli, Capelli, Monet and Senellar, 2020; $\cdots$]

Thus, if the support of a function f coincides with the true points of a Boolean function that admits only exponential size DNNF circuits, then f admits only exponential size decomposable AC that use only positive weights.

Reduction to Tractable Boolean Circuits – the Boring Trick

This works only for AC with positive weights.

Note that when the AC is deterministic, the weights can always be taken positive with no impact on the size of the AC.



Also, for deterministic decomposable AC with integer weights, there is a poly-time transformation to a d-DNNF circuit computing $\mathbb{1}[C(X) \geq k]$ for any constant k (not just 0) while this is generally intractable for decomposable AC.

Roadmap

- ~~Tractable arithmetic circuits (AC)~~
- ~~Reduction to tractable Boolean circuits (the boring trick)~~
- Lower bounds via the rank technique
- Separating classes of tractable ACs

Lower Bounds via the Rank Technique

The rank technique allows one to prove lower bounds on the size of structured-decomposable AC (possibly using negative weights) for a function F.

We use the notion of value matrix for a function F.

Definition (Value Matrix)

Let $F(\boldsymbol{X})$ be a function and $\Pi = (\boldsymbol{A}, \boldsymbol{B})$ be a partition of $\boldsymbol{X}$. A value matrix $M_\Pi(F)$ is a $2^{|\boldsymbol{B}|} \times 2^{|\boldsymbol{A}|}$ matrix whose columns are indexed by the complete assignments to $\boldsymbol{A}$, whose rows are indexed by the complete assignments to $\boldsymbol{B}$ and such that the entry at column $\vec{X}_{\boldsymbol{A}}$ and row $\vec{X}_{\boldsymbol{B}}$ is $F(\vec{X}_{\boldsymbol{A}}, \vec{X}_{\boldsymbol{B}})$.

Lower Bounds via the Rank Technique

Example: $F(X_1, X_2, X_3, X_4, X_5) = \frac{1}{2}(X_1 + X_5)(X_2 + 3X_4) + X_2X_3$ and $\Pi = (X_1X_2X_3, X_4X_5)$. A value matrix $M_\Pi(F)$ is

$$
\begin{array}{cc}
 & \begin{array}{ccccccccc} X_1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ X_2 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ X_3 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{array} \\
\begin{array}{c} X_4X_5 \\ 0\ 0 \\ 1\ 0 \\ 0\ 1 \\ 1\ 1 \end{array} & \begin{pmatrix} 0 & 0 & 1 & 0 & \frac{3}{2} & 0 & \frac{3}{2} & \frac{1}{2} \\ 0 & \frac{1}{2} & 1 & 0 & 3 & \frac{3}{2} & 3 & 2 \\ 0 & 0 & \frac{3}{2} & 0 & 2 & 0 & 2 & 1 \\ \frac{3}{2} & 3 & 3 & \frac{3}{2} & 5 & 3 & 5 & 4 \end{pmatrix}
\end{array}
$$

The order of the rows and the order of the columns is not important for us.

Lower Bounds via the Rank Technique

Example: $F(X_1, X_2, X_3, X_4, X_5) = \frac{1}{2}(X_1 + X_5)(X_2 + 3X_4) + X_2X_3$ and $\Pi = (X_1X_2X_3, X_4X_5)$. A value matrix $M_\Pi(F)$ is

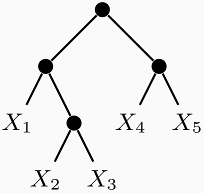
$$
\begin{array}{cc}
 & \begin{array}{cccccccc} X_1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ X_2 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ X_3 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{array} \\
\begin{array}{c} X_4X_5 \\ 0\ 0 \\ 1\ 0 \\ 0\ 1 \\ 1\ 1 \end{array} &
\begin{pmatrix} 0 & 0 & 1 & 0 & \frac{3}{2} & 0 & \frac{3}{2} & \frac{1}{2} \\ 0 & \frac{1}{2} & 1 & 0 & 3 & \frac{3}{2} & 3 & 2 \\ 0 & 0 & \frac{3}{2} & 0 & 2 & 0 & 2 & 1 \\ \frac{3}{2} & 3 & 3 & \frac{3}{2} & 5 & 3 & 5 & 4 \end{pmatrix}
\end{array}
\longrightarrow F(0,1,1,0,1) = 2
$$

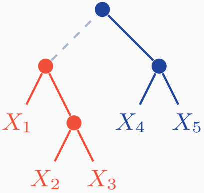
The order of the rows and the order of the columns is not important for us.

Lower Bounds via the Rank Technique

Theorem

Let T be a vtree and let $\Pi = (\boldsymbol{A}, \boldsymbol{B})$ of $\boldsymbol{X}$ be a partition induced by T. Every structured-decomposable AC respecting T and representing $F(\boldsymbol{X})$ contains at least $\mathsf{rank}(M_\Pi(F))$ *gates.*





Corollary

Every structured-decomposable AC representing $F(\boldsymbol{X})$ contains at least

$$\min_{\Pi} \mathsf{rank}(M_\Pi(F))$$

gates where Π ranges over all ***balanced*** *partitions (i.e., $\frac{|\boldsymbol{X}|}{3} \leq |\boldsymbol{A}|, |\boldsymbol{B}| \leq \frac{2|\boldsymbol{X}|}{3}$).*

Lower Bounds via the Rank Technique

Example of “hard” functions: consider a simple graphs G and let

$$F_G(\boldsymbol{X}) = \prod_{(i,j)\in E(G)} (\,1 + \max(X_i, X_j)\,).$$

Note that F_G never takes value 0.

Theorem

If G is a (c,d)-expander graph with $d = O(1)$, then for all balanced partitions Π of $\boldsymbol{X}$,

$$\mathit{rank}(M_\Pi(F_G(\boldsymbol{X}))) \geq 2^{\Omega(n)}$$

where $n = |\boldsymbol{X}|$.

Corollary

There exists an infinite class of (c,3)-expander graphs for some constant c, so there exists an infinite class of functions $F_G(\boldsymbol{X})$ computed only by structured-decomposable AC of size $2^{\Omega(n)}$, where $n = |\boldsymbol{X}|$.

Lower Bounds via the Rank Technique

The proof of the rank theorem uses a decomposition of the AC as sum of "arithmetic rectangles" (for lack of better term)

$$\begin{tikzcd} \boxed{\begin{array}{c}\text{str.-dec.}\\ \text{AC } C\end{array}} \arrow[r, "\text{circuit}", "\text{analysis}"'] & \boxed{\begin{array}{c} C \equiv \sum_{i=1}^{k} \overbrace{f_i(A) \cdot g_i(B)}^{P_i(A,B)}, \\ (A,B) \text{ balanced, } |C| \geq k \end{array}} \arrow[r, "M_\Pi(P_i)", "\text{has rank } 1"'] & \boxed{\begin{array}{c} |C| \geq k \geq \\ \text{rank}(M_\Pi(C)) \end{array}} \end{tikzcd}$$

This is similar to the lower bound techniques for DNNF circuits using combinatorial rectangles.

Roadmap

- ~~Tractable arithmetic circuits (AC)~~
- ~~Reduction to tractable Boolean circuits (the boring trick)~~
- ~~Lower bounds via the rank technique~~
- Separating classes of tractable ACs

Separating Classes of Tractable ACs

Showing a separation between two classes of tractable AC = finding an infinite class of functions that admit polynomial-size representations in one class but only super-polynomial-size representations in the other.

Building on the previous result, we can show the separation between the class of decomposable ACs and that of structured-decomposable ACs.

Theorem (dec. AC exponentially separated from str-dec. ACs)

There is an infinite class of functions that have small decomposable ACs but only exponential-size structured-decomposable ACs.

Separating Classes of Tractable ACs

We just need the following results...

Lemma (Conditioning str-dec. ACs)

Conditioning is done in linear-time on str-dec. AC and preserve structured-decomposability.

$$C(\boldsymbol{X}) \xrightarrow[\textit{linear-time}]{} C'(\boldsymbol{X} \setminus var(\alpha)) \equiv C(\boldsymbol{X})|\alpha$$

where α is a partial assignment to $\boldsymbol{X}$.

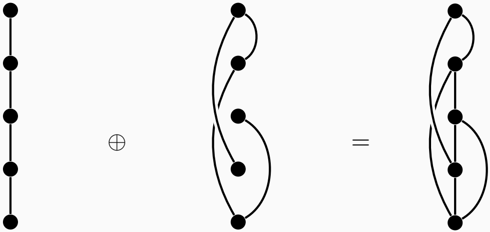
Lemma (Conjunction of str-dec. ACs)

Taking the product of two str-dec. AC while preserving structured-decomposability is tractable when they respect the same vtree.

$$C_1(\boldsymbol{X}), C_2(\boldsymbol{X}) \xrightarrow[\textit{quadratic-time}]{} C(\boldsymbol{X}) \equiv C_1(\boldsymbol{X}) \cdot C_2(\boldsymbol{X})$$

[Vergari, Choi, Liu, Teso and Van den Broeck, 2021]

...and some knowledge about expander graphs.



G_1 = fixed path $\quad$ G_2 = random path $\quad$ $G_1 \oplus G_2$

Lemma

When G_2 is chosen uniformly at random, $G_1 \oplus G_2$ is an expander graph with high probability.

Separating Classes of Tractable ACs

1. With high probability $F_{G_1 \oplus G_2}(\boldsymbol{X})$ admits only exponential-size structured-decomposable ACs

Separating Classes of Tractable ACs

1. With high probability $F_{G_1 \oplus G_2}(\boldsymbol{X})$ admits only exponential-size structured-decomposable ACs

2. But $F_{G_1 \oplus G_2}(\boldsymbol{X}) = F_{G_1'}(\boldsymbol{X}) \cdot F_{G_2}(\boldsymbol{X})$ (for G_1' a subgraph of G_1)

Separating Classes of Tractable ACs

1. With high probability $F_{G_1 \oplus G_2}(\boldsymbol{X})$ admits only exponential-size structured-decomposable ACs
2. But $F_{G_1 \oplus G_2}(\boldsymbol{X}) = F_{G_1'}(\boldsymbol{X}) \cdot F_{G_2}(\boldsymbol{X})$ (for G_1' a subgraph of G_1)
3. So for every vtree T, $F_{G_1'}(\boldsymbol{X})$ or $F_{G_2}(\boldsymbol{X})$ admits only exponential-size structured-decomposable ACs respecting T.

Separating Classes of Tractable ACs

1. With high probability $F_{G_1 \oplus G_2}(\boldsymbol{X})$ admits only exponential-size structured-decomposable ACs
2. But $F_{G_1 \oplus G_2}(\boldsymbol{X}) = F_{G_1'}(\boldsymbol{X}) \cdot F_{G_2}(\boldsymbol{X})$ (for G_1' a subgraph of G_1)
3. So for every vtree T, $F_{G_1'}(\boldsymbol{X})$ or $F_{G_2}(\boldsymbol{X})$ admits only exponential-size structured-decomposable ACs respecting T.
4. Both $F_{G_1'}(\boldsymbol{X})$ and $F_{G_2}(\boldsymbol{X})$ has small structured-decomposable AC for different vtrees (because G_1 and G_2 are paths)

Separating Classes of Tractable ACs

1. With high probability $F_{G_1 \oplus G_2}(\boldsymbol{X})$ admits only exponential-size structured-decomposable ACs
2. But $F_{G_1 \oplus G_2}(\boldsymbol{X}) = F_{G'_1}(\boldsymbol{X}) \cdot F_{G_2}(\boldsymbol{X})$ (for G'_1 a subgraph of G_1)
3. So for every vtree T, $F_{G'_1}(\boldsymbol{X})$ or $F_{G_2}(\boldsymbol{X})$ admits only exponential-size structured-decomposable ACs respecting T.
4. Both $F_{G'_1}(\boldsymbol{X})$ and $F_{G_2}(\boldsymbol{X})$ has small structured-decomposable AC for different vtrees (because G_1 and G_2 are paths)
5. Let y be a fresh variable, then $F(\boldsymbol{X}, y) := (y \cdot F_{G'_1}(\boldsymbol{X})) + (\neg y \cdot F_{G_2}(\boldsymbol{X}))$ has small decomposable AC.

Separating Classes of Tractable ACs

1. With high probability $F_{G_1 \oplus G_2}(\boldsymbol{X})$ admits only exponential-size structured-decomposable ACs
2. But $F_{G_1 \oplus G_2}(\boldsymbol{X}) = F_{G_1'}(\boldsymbol{X}) \cdot F_{G_2}(\boldsymbol{X})$ (for G_1' a subgraph of G_1)
3. So for every vtree T, $F_{G_1'}(\boldsymbol{X})$ or $F_{G_2}(\boldsymbol{X})$ admits only exponential-size structured-decomposable ACs respecting T.
4. Both $F_{G_1'}(\boldsymbol{X})$ and $F_{G_2}(\boldsymbol{X})$ has small structured-decomposable AC for different vtrees (because G_1 and G_2 are paths)
5. Let y be a fresh variable, then $F(\boldsymbol{X}, y) := (y \cdot F_{G_1'}(\boldsymbol{X})) + (\neg y \cdot F_{G_2}(\boldsymbol{X}))$ has small decomposable AC.
6. If F had a small structured-decomposable AC respecting some T then $F|(y = 1)$ and $F|(y = 0)$ would both have small structured-decomposable AC respecting T, which is not possible due to 3.

Separating Classes of Tractable ACs

So this proves the separation of structured-decomposable ACs and decomposable ACs.

Theorem (dec. AC exponentially separated from str-dec. ACs)

There is an infinite class of functions that have small decomposable ACs but only exponential-size structured-decomposable ACs.

Again, we have made no assumption on the sign of the weights used in our ACs.

Roadmap

- ~~Tractable arithmetic circuits (AC)~~
- ~~Reduction to tractable Boolean circuits (the boring trick)~~
- ~~Lower bounds via the rank technique~~
- ~~Separating classes of tractable ACs~~

Thank you