Approximating Polynomials for De Morgan Formulas with Optimal Coefficient ℓ_1-Norm Bounds

Yichuan Wang

UC Berkeley

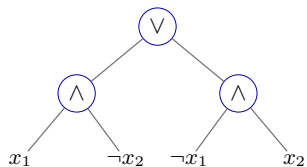
STOC video

De Morgan Formulas

De Morgan Formula: a binary tree of AND / OR gates (no XOR gates), with literals at the leaves.

$$F : \{0,1\}^n \longrightarrow \{0,1\}.$$

- n: number of input bits.
- S: formula size, counting number of leaves.

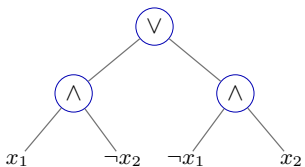


$n = 2$ input bits, $S = 4$ leaves

De Morgan Formulas

De Morgan Formula: a binary tree of AND / OR gates (no XOR gates), with literals at the leaves.

$$F : \{0,1\}^n \longrightarrow \{0,1\}.$$



$n = 2$ input bits, $S = 4$ leaves

- n: number of input bits.
- S: formula size, counting number of leaves.

Polynomial-size formulas capture nonuniform NC^1.

Technique: real polynomial approximation.

Previous Result and Our Theorem

Fix an S-leaf De Morgan formula $F : \{0,1\}^n \to \{0,1\}$.

Previous result [Ambainis et al. '10; Reichardt '11]

Exists a real polynomial P that $1/3$-approximates F pointwise, with $\deg P = O(\sqrt{S})$.

Pointwise Approximate: $|P(x) - F(x)| \leq 1/3$ for **every** $x \in \{0,1\}^n$.

Previous Result and Our Theorem

Fix an S-leaf De Morgan formula $F : \{0,1\}^n \to \{0,1\}$.

Previous result [Ambainis et al. '10; Reichardt '11]

Exists a real polynomial P that 1/3-approximates F pointwise, with $\deg P = O(\sqrt{S})$.

Pointwise Approximate: $|P(x) - F(x)| \leq 1/3$ for **every** $x \in \{0,1\}^n$.

Our theorem

Exists a real polynomial P that 1/3-approximates F pointwise, with $\deg P = O(\sqrt{S})$ and $\|P\|_1 \leq 2^{O(\sqrt{S})}$.

Coefficient ℓ_1-norm: for $P(x) = \sum_{T \subseteq [n]} a_T \prod_{i \in T} x_i$, $a_T \in \mathbb{R}$,
define $\|P\|_1 = \sum_{T \subseteq [n]} |a_T|$ in the $\{0,1\}$-monomial basis.

Previous Result and Our Theorem

Fix an S-leaf De Morgan formula $F : \{0,1\}^n \to \{0,1\}$.

Previous result [Ambainis et al. '10; Reichardt '11]

Exists a real polynomial P that 1/3-approximates F pointwise, with $\deg P = O(\sqrt{S})$.

Pointwise Approximate: $|P(x) - F(x)| \leq 1/3$ for **every** $x \in \{0,1\}^n$.

Our theorem

Exists a real polynomial P that 1/3-approximates F pointwise, with $\deg P = O(\sqrt{S})$ and $\|P\|_1 \leq 2^{O(\sqrt{S})}$.

Coefficient ℓ_1-norm: for $P(x) = \sum_{T \subseteq [n]} a_T \prod_{i \in T} x_i$, $a_T \in \mathbb{R}$,
define $\|P\|_1 = \sum_{T \subseteq [n]} |a_T|$ in the $\{0,1\}$-monomial basis.

Direct estimate: degree-$O(\sqrt{S})$ gives $\|P\|_1 \leq 2^{O(\sqrt{S} \log S)}$.

Previous Result and Our Theorem

Fix an S-leaf De Morgan formula $F : \{0,1\}^n \to \{0,1\}$.

Previous result [Ambainis et al. '10; Reichardt '11]

Exists a real polynomial P that 1/3-approximates F pointwise, with $\deg P = O(\sqrt{S})$.

Pointwise Approximate: $|P(x) - F(x)| \leq 1/3$ for **every** $x \in \{0,1\}^n$.

Our theorem

Exists a real polynomial P that 1/3-approximates F pointwise, with $\deg P = O(\sqrt{S})$ and $\|P\|_1 \leq 2^{O(\sqrt{S})}$.

Coefficient ℓ_1-norm: for $P(x) = \sum_{T \subseteq [n]} a_T \prod_{i \in T} x_i$, $a_T \in \mathbb{R}$,
define $\|P\|_1 = \sum_{T \subseteq [n]} |a_T|$ in the $\{0,1\}$-monomial basis.

Direct estimate: degree-$O(\sqrt{S})$ gives $\|P\|_1 \leq 2^{O(\sqrt{S} \log S)}$.

For the proof: it suffices to take $S = n$, with one distinct input variable per leaf.

Applications of the Coefficient ℓ_1-Norm

Coefficient ℓ_1 bounds are used in formula lower bounds and pseudorandom generators.
[Tal '17; Hatami–Hoza–Tal–Tell '21; Bathie–Williams '24; ...]

Applications of the Coefficient ℓ_1-Norm

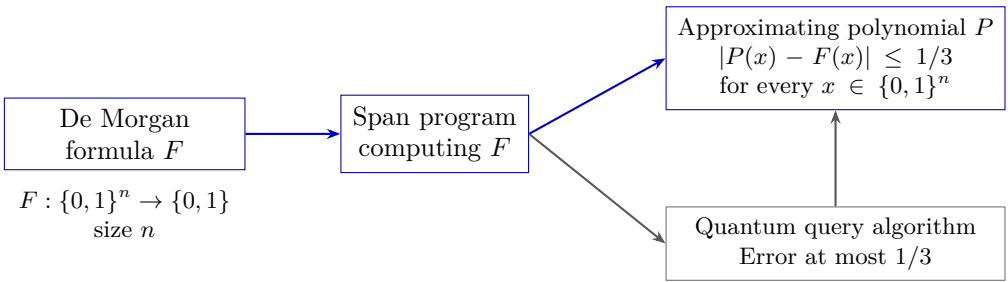
Coefficient ℓ_1 bounds are used in formula lower bounds and pseudorandom generators. [Tal '17; Hatami–Hoza–Tal–Tell '21; Bathie–Williams '24; ...]

Application: A De Morgan formula-size lower bound for an explicit function in P, on n input bits.

Previous bound [Tal '17]	With our theorem
$\Omega\left(\frac{n^3}{\log n\,(\log\log n)^2}\right)$	$\Omega\left(\frac{n^3}{\log n\,\log\log n}\right)$

A factor of $\log\log n$ improvement.

Proof Roadmap



Definition of Span Programs

Input: $x \in \{0,1\}^n$.

Description of a Span Program: Unit vectors $\boldsymbol{v}_1, \cdots, \boldsymbol{v}_m \in \mathbb{R}^d$ *(vector list)*, and unit vector $\boldsymbol{\tau} \in \mathbb{R}^d$ *(target vector)*; each $\boldsymbol{v}_j$ has one of the following $2n+1$ labels

$$\text{always available}, \quad x_1, \cdots, x_n, \quad \neg x_1, \cdots, \neg x_n.$$

(Usually $m, d \approx \text{poly}(n)$)

Definition of Span Programs

Input: $x \in \{0,1\}^n$.

Description of a Span Program: Unit vectors $\boldsymbol{v}_1, \cdots, \boldsymbol{v}_m \in \mathbb{R}^d$ *(vector list)*, and unit vector $\boldsymbol{\tau} \in \mathbb{R}^d$ *(target vector)*; each $\boldsymbol{v}_j$ has one of the following $2n+1$ labels

$$\text{always available}, \quad x_1, \cdots, x_n, \quad \neg x_1, \cdots, \neg x_n.$$

(Usually $m, d \approx \text{poly}(n)$)

Behavior on input x: *Available vectors on x* are exactly those labelled always available, x_i with $x_i = 1$, or $\neg x_i$ with $x_i = 0$. Let $I_x \subseteq [m]$ be *available vectors*' indices.

Output 1: $\boldsymbol{\tau} \in \text{span}\{\boldsymbol{v}_j : j \in I_x\}$

A 1-**witness**: real coefficients a_j with

$$\sum_{j \in I_x} a_j \boldsymbol{v}_j = \boldsymbol{\tau}.$$

Its size is $\sum_{j \in I_x} a_j^2$.

Output 0: $\boldsymbol{\tau} \notin \text{span}\{\boldsymbol{v}_j : j \in I_x\}$

A 0-**witness**: $\boldsymbol{w} \in \mathbb{R}^d$ with

$$\langle \boldsymbol{w}, \boldsymbol{\tau} \rangle = 1,$$
$$\langle \boldsymbol{w}, \boldsymbol{v}_j \rangle = 0 \quad (j \in I_x).$$

Its size is $\|\boldsymbol{w}\|_2^2$.

Definition of Span Programs

Input: $x \in \{0,1\}^n$.

Description of a Span Program: Unit vectors $\boldsymbol{v}_1, \cdots, \boldsymbol{v}_m \in \mathbb{R}^d$ *(vector list)*, and unit vector $\boldsymbol{\tau} \in \mathbb{R}^d$ *(target vector)*; each $\boldsymbol{v}_j$ has one of the following $2n+1$ labels

$$\text{always available}, \quad x_1, \cdots, x_n, \quad \neg x_1, \cdots, \neg x_n.$$

(Usually $m, d \approx \text{poly}(n)$)

Behavior on input x: *Available vectors on x* are exactly those labelled always available, x_i with $x_i = 1$, or $\neg x_i$ with $x_i = 0$. Let $I_x \subseteq [m]$ be *available vectors*' indices.

Output 1: $\boldsymbol{\tau} \in \text{span}\{\boldsymbol{v}_j : j \in I_x\}$

A 1-**witness**: real coefficients a_j with

$$\sum_{j \in I_x} a_j \boldsymbol{v}_j = \boldsymbol{\tau}.$$

Its size is $\sum_{j \in I_x} a_j^2$.

Output 0: $\boldsymbol{\tau} \notin \text{span}\{\boldsymbol{v}_j : j \in I_x\}$

A 0-**witness**: $\boldsymbol{w} \in \mathbb{R}^d$ with

$$\langle \boldsymbol{w}, \boldsymbol{\tau} \rangle = 1,$$
$$\langle \boldsymbol{w}, \boldsymbol{v}_j \rangle = 0 \quad (j \in I_x).$$

Its size is $\|\boldsymbol{w}\|_2^2$.

Complexity Measure: witness size W: every input has a witness of size at most W.

Example: A Span Program for $x_1 \wedge x_2$

Space: $\mathbb{R}^2$.

$$\boldsymbol{v}_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \text{ (label } x_1\text{)}, \qquad \boldsymbol{v}_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \text{ (label } x_2\text{)}, \qquad \boldsymbol{\tau} = \frac{1}{\sqrt{2}} \cdot \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

$\boldsymbol{v}_i$ is available when $x_i = 1$. Accept if $\boldsymbol{\tau}$ is in the span of the available vectors.

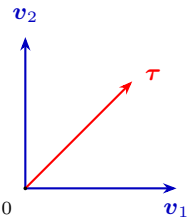
Example: A Span Program for $x_1 \wedge x_2$

Space: $\mathbb{R}^2$.

$$\boldsymbol{v}_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \text{ (label } x_1\text{)}, \qquad \boldsymbol{v}_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \text{ (label } x_2\text{)}, \qquad \boldsymbol{\tau} = \frac{1}{\sqrt{2}} \cdot \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

$\boldsymbol{v}_i$ is available when $x_i = 1$. Accept if $\boldsymbol{\tau}$ is in the span of the available vectors.

Input $(1, 1)$**: output** 1



$\boldsymbol{\tau} = \dfrac{\boldsymbol{v}_1 + \boldsymbol{v}_2}{\sqrt{2}}$. Witness size $= 1$.

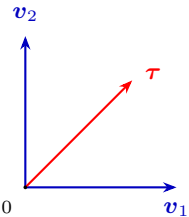
Example: A Span Program for $x_1 \wedge x_2$

Space: $\mathbb{R}^2$.

$$\boldsymbol{v}_1 = \begin{pmatrix}1\\0\end{pmatrix} \text{ (label } x_1\text{)}, \qquad \boldsymbol{v}_2 = \begin{pmatrix}0\\1\end{pmatrix} \text{ (label } x_2\text{)}, \qquad \boldsymbol{\tau} = \frac{1}{\sqrt{2}} \cdot \begin{pmatrix}1\\1\end{pmatrix}.$$

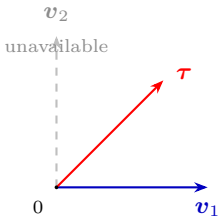
$\boldsymbol{v}_i$ is available when $x_i = 1$. Accept if $\boldsymbol{\tau}$ is in the span of the available vectors.

Input $(1,1)$: output 1



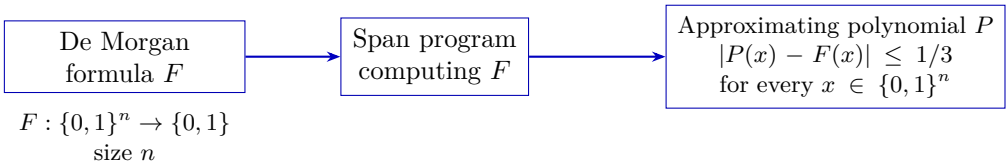
$\boldsymbol{\tau} = \dfrac{\boldsymbol{v}_1 + \boldsymbol{v}_2}{\sqrt{2}}$. Witness size $= 1$.

Input $(1,0)$: output 0

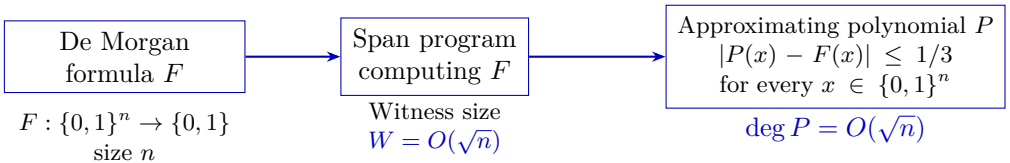


$\boldsymbol{\tau} \notin \text{span}\{\boldsymbol{v}_1\}$.
Witness: $\boldsymbol{w} = \sqrt{2} \cdot \boldsymbol{v}_2$; Size: 2.

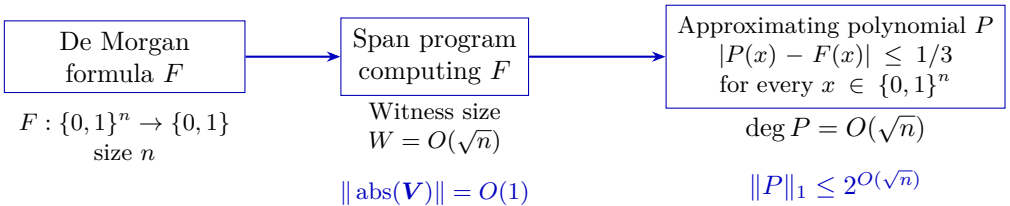
Proof Roadmap



Proof Roadmap



Proof Roadmap

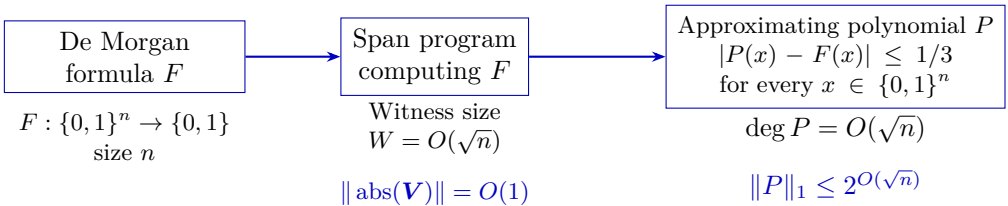


Proof Roadmap

$\boldsymbol{V} = \begin{pmatrix} | & & | \\ \boldsymbol{v}_1 & \cdots & \boldsymbol{v}_m \\ | & & | \end{pmatrix} \in \mathbb{R}^{d \times m}$ contains **all listed vectors**.

$\mathrm{abs}(\boldsymbol{V})$ takes absolute values entry by entry.

$\|\cdot\|$ denotes the Euclidean matrix norm: maximum of $\boldsymbol{x}^\top \boldsymbol{V} \boldsymbol{y}$ over unit vectors $\boldsymbol{x}, \boldsymbol{y}$.

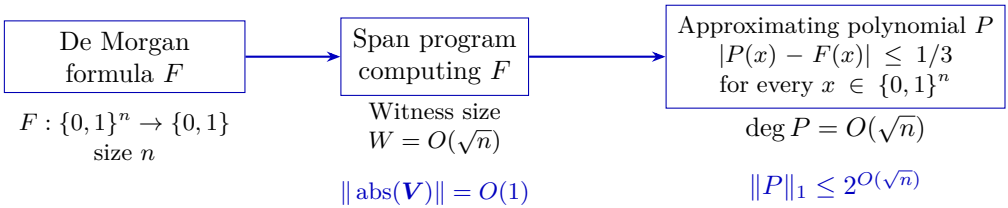


Proof Roadmap

$\boldsymbol{V} = \begin{pmatrix} | & & | \\ \boldsymbol{v}_1 & \cdots & \boldsymbol{v}_m \\ | & & | \end{pmatrix} \in \mathbb{R}^{d \times m}$ contains **all listed vectors**.

$\mathrm{abs}(\boldsymbol{V})$ takes absolute values entry by entry.

$\|\cdot\|$ denotes the Euclidean matrix norm: maximum of $\boldsymbol{x}^\top \boldsymbol{V} \boldsymbol{y}$ over unit vectors $\boldsymbol{x}, \boldsymbol{y}$.



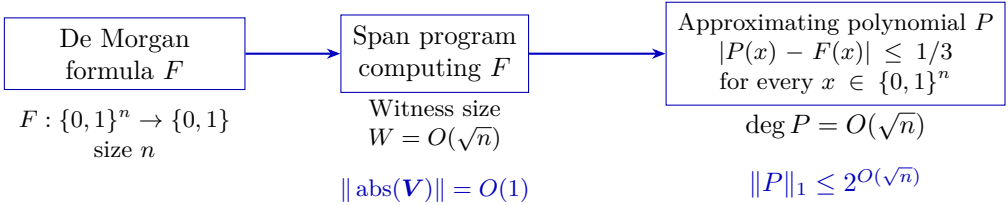
"Span program → Polynomial" proof idea: use matrix powering to estimate the eigenvalue behavior of $\boldsymbol{V}$ (set unavailable columns to 0).

Proof Roadmap

$\boldsymbol{V} = \begin{pmatrix} | & & | \\ \boldsymbol{v}_1 & \cdots & \boldsymbol{v}_m \\ | & & | \end{pmatrix} \in \mathbb{R}^{d \times m}$ contains **all listed vectors**.

$\mathrm{abs}(\boldsymbol{V})$ takes absolute values entry by entry.

$\|\cdot\|$ denotes the Euclidean matrix norm: maximum of $\boldsymbol{x}^\top \boldsymbol{V} \boldsymbol{y}$ over unit vectors $\boldsymbol{x}, \boldsymbol{y}$.



"Span program → Polynomial" proof idea: use matrix powering to estimate the eigenvalue behavior of $\boldsymbol{V}$ (set unavailable columns to 0).

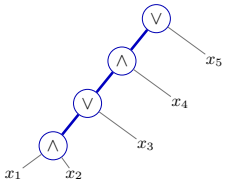
Our Main Contribution: a new construction of "Formulas → Span program" that guarantees $\|\mathrm{abs}(\boldsymbol{V})\| = O(1)$.

Two Partial Attempts

Goal: $W = O(\sqrt{n})$ and $\|\operatorname{abs}(\boldsymbol{V})\| = O(1)$.

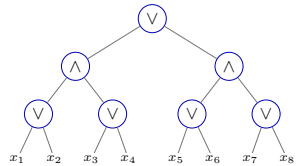
Attempt I

(*Gate Composition*, by the human author)



Attempt II

(*Helper Dimension*, by GPT-5.5 Pro)



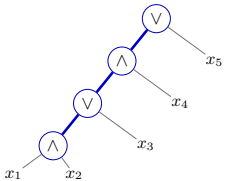
Adapted from ideas of Ambainis et al. and Reichardt; see the paper for details.

Two Partial Attempts

Goal: $W = O(\sqrt{n})$ and $\|\operatorname{abs}(\boldsymbol{V})\| = O(1)$.

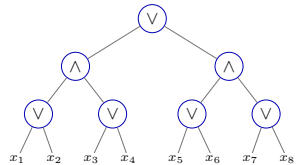
Attempt I

(*Gate Composition*, by the human author)



Attempt II

(*Helper Dimension*, by GPT-5.5 Pro)



Obtain $W = O(\sqrt{n})$ for any formula,

but $\|\operatorname{abs}(\boldsymbol{V})\| = O(1)$ is guaranteed only for chains.

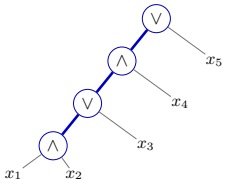
Adapted from ideas of Ambainis et al. and Reichardt; see the paper for details.

Two Partial Attempts

Goal: $W = O(\sqrt{n})$ and $\|\operatorname{abs}(\boldsymbol{V})\| = O(1)$.

Attempt I

(*Gate Composition*, by the human author)

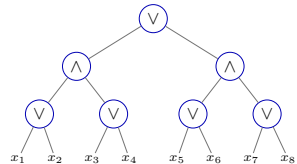


Obtain $W = O(\sqrt{n})$ for any formula,

but $\|\operatorname{abs}(\boldsymbol{V})\| = O(1)$ is guaranteed only for chains.

Attempt II

(*Helper Dimension*, by GPT-5.5 Pro)

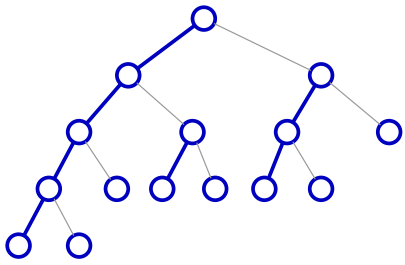


Achieve $W = O(\sqrt{n})$ and $\|\operatorname{abs}(\boldsymbol{V})\| = O(1)$ for balanced trees

But for general cases (e.g. a chain) even $W = O(\sqrt{n})$ can fail.

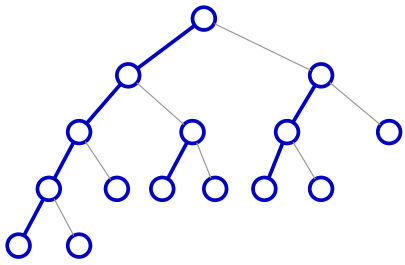
Adapted from ideas of Ambainis et al. and Reichardt; see the paper for details.

Arbitrary Formula: Heavy-Chain Decomposition



Thick blue edges form the chains.

Arbitrary Formula: Heavy-Chain Decomposition

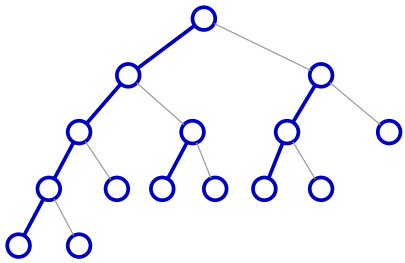


Thick blue edges form the chains.

Within each chain:
use the gate-composition rule
from Attempt I.

To combine the chains:
use the helper-dimension rule
from Attempt II.

Arbitrary Formula: Heavy-Chain Decomposition



Thick blue edges form the chains.

Within each chain:
use the gate-composition rule
from Attempt I.

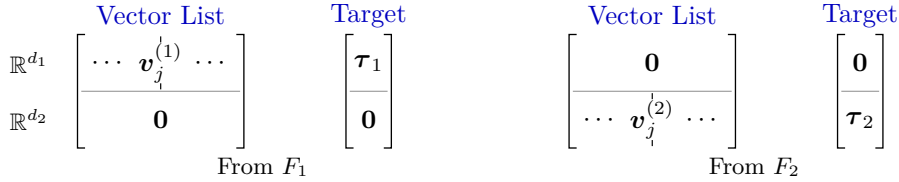
To combine the chains:
use the helper-dimension rule
from Attempt II.

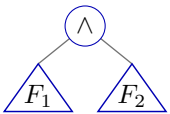
Both bounds for every formula of size n

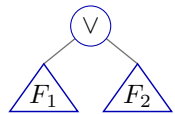
$$W = O(\sqrt{n}), \qquad \|\operatorname{abs}(\boldsymbol{V})\| = O(1).$$

Component I: Gate Composition

Goal: merge two child span programs for subtrees F_1, F_2.

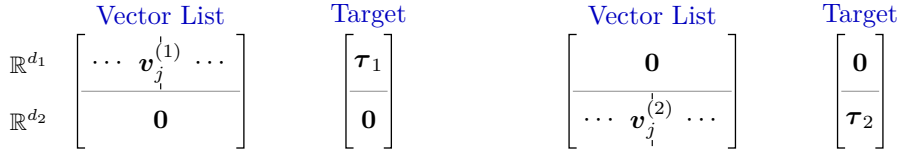




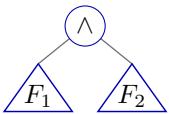


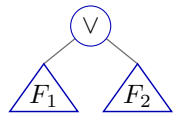
Component I: Gate Composition

Goal: merge two child span programs for subtrees F_1, F_2.



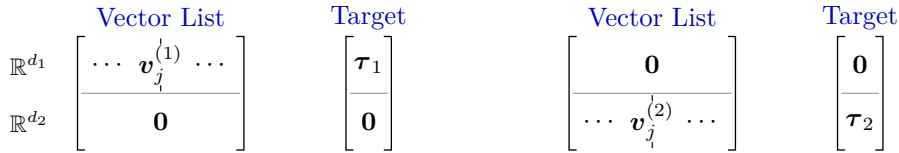
New target: $\boldsymbol{\tau} := \frac{1}{\sqrt{2}} \begin{bmatrix} \boldsymbol{\tau}_1 \\ \boldsymbol{\tau}_2 \end{bmatrix}$



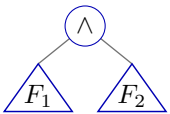


Component I: Gate Composition

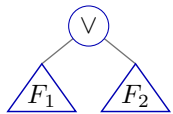
Goal: merge two child span programs for subtrees F_1, F_2.



$$\text{New target: } \boldsymbol{\tau} := \frac{1}{\sqrt{2}} \begin{bmatrix} \boldsymbol{\tau}_1 \\ \boldsymbol{\tau}_2 \end{bmatrix}$$



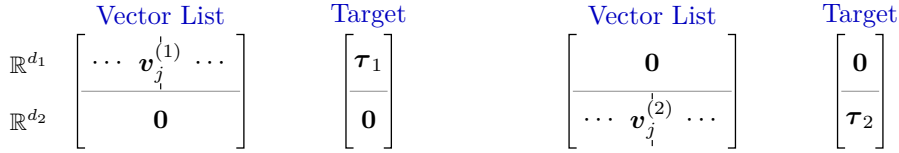
Keep vector lists from both childs



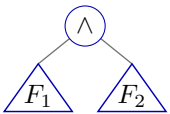
If both children accept, their targets generate the parent target $\boldsymbol{\tau}$.

Component I: Gate Composition

Goal: merge two child span programs for subtrees F_1, F_2.

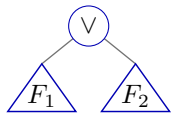


$$\text{New target: } \boldsymbol{\tau} := \frac{1}{\sqrt{2}} \begin{bmatrix} \boldsymbol{\tau}_1 \\ \boldsymbol{\tau}_2 \end{bmatrix}$$



Keep vector lists from both childs

If both children accept, their targets generate the parent target $\boldsymbol{\tau}$.



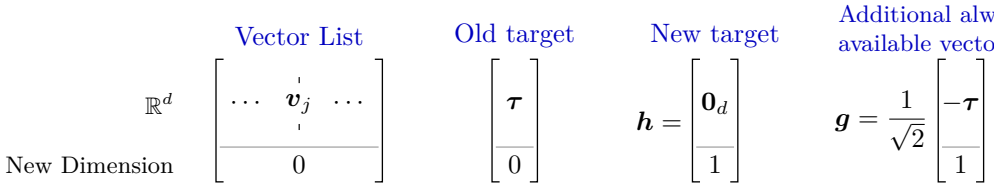
$$\boldsymbol{r} = \frac{1}{\sqrt{2}} \begin{bmatrix} \boldsymbol{\tau}_1 \\ -\boldsymbol{\tau}_2 \end{bmatrix}$$

Additional always available vector

If either child accepts, its target and $\boldsymbol{r}$ generate the parent target $\boldsymbol{\tau}$.

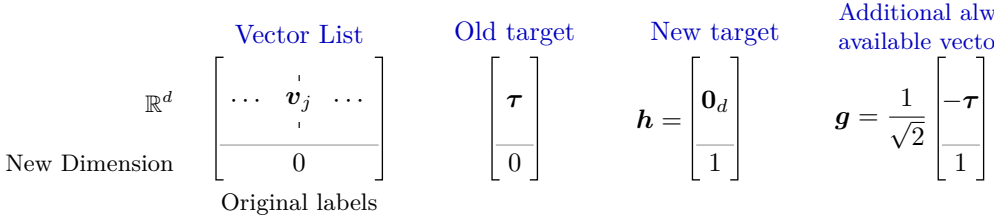
Component II: Helper Dimension

Start with a span program in $\mathbb{R}^d$, with unit target $\boldsymbol{\tau}$.
Goal: replace the target by a standard basis vector (i.e., $(0, \cdots, 0, 1)$)



Component II: Helper Dimension

Start with a span program in $\mathbb{R}^d$, with unit target $\boldsymbol{\tau}$.
Goal: replace the target by a standard basis vector (i.e., $(0, \cdots, 0, 1)$)



The output is unchanged:

$$\boldsymbol{h} = \begin{pmatrix} \boldsymbol{\tau} \\ 0 \end{pmatrix} + \sqrt{2}\,\boldsymbol{g}.$$

The new program accepts if and only if the old program accepts.

Summary

Main result

Every S-leaf De Morgan formula $F : \{0,1\}^n \to \{0,1\}$ has a real polynomial P such that

$$|P(x) - F(x)| \leq \tfrac{1}{3} \qquad \text{for every } x \in \{0,1\}^n,$$

$$\deg P = O(\sqrt{S}), \qquad \|P\|_1 \leq 2^{O(\sqrt{S})}.$$

Span-program construction:

- Two partial attempts: gate composition for long chains, and helper dimension for balanced trees.
- Heavy-chain decomposition combines them for general formulas.

Thank you!

Summary

Main result

Every S-leaf De Morgan formula $F : \{0,1\}^n \to \{0,1\}$ has a real polynomial P such that

$$|P(x) - F(x)| \leq \tfrac{1}{3} \qquad \text{for every } x \in \{0,1\}^n,$$

$$\deg P = O(\sqrt{S}), \qquad \|P\|_1 \leq 2^{O(\sqrt{S})}.$$

Span-program construction:

- Two partial attempts: gate composition for long chains, and helper dimension for balanced trees.
- Heavy-chain decomposition combines them for general formulas.

Thank you!