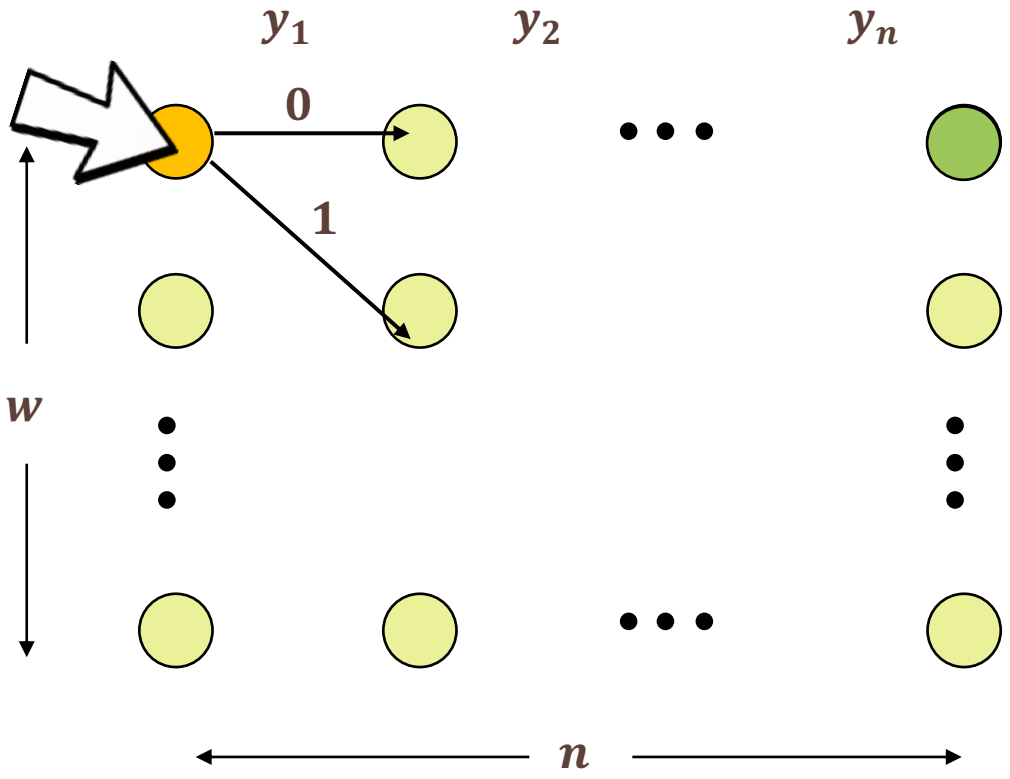
Pseudorandom Pseudo-Distributions for ROBPs with Optimal Dependence on Error

Sumegha Garg

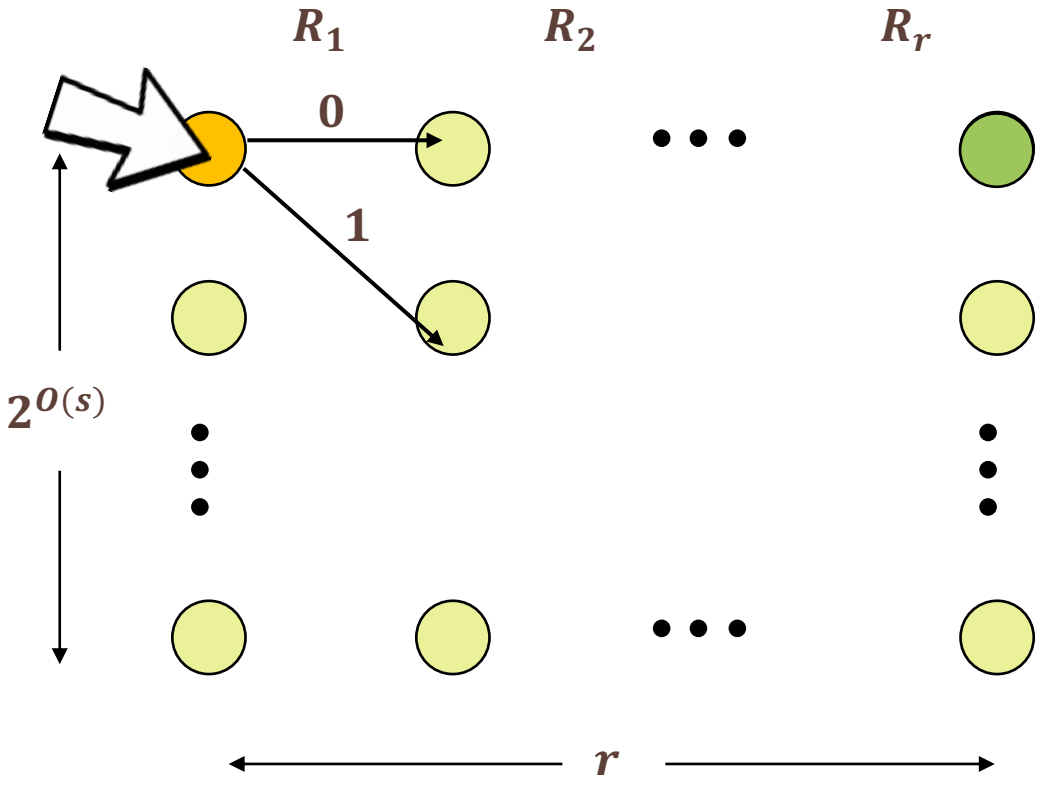
Read-Once Branching Programs (ROBPs)

Can we estimate the probability of acceptance for every width-w length-n ROBP?



Read-Once Branching Programs (ROBPs)

If M uses space s and r random bits. then running of M on input x can be simulated by width-$2^{O(s)}$ length-r ROBP.



Pseudorandom Generator (PRG)

A (w, n, ϵ)-PRG is a function $G: \{0,1\}^l \to \{0,1\}^n$ such that for every width w length n ROBP B,

$$\Pr_{x \sim \{0,1\}^l}[B(G(x)) = 1] \approx_\epsilon \Pr_{y \sim \{0,1\}^n}[B(y) = 1]$$

The l-bit input is the seed.

Definition of Explicit

Given l-bit input x and an index $i \in [n]$, ith bit of $G(x)$ can be computed in $O(l + \log n)$ space.

$\boldsymbol{l = O(\log n)}$ **solves** $\boldsymbol{BPL = L}$ (log-space computable G; $w = n$)

Hitting Set

A (w, n, ϵ)-hitting set is a set $S \subseteq \{0,1\}^n$ such that, for every width w length n ROBP B, for which,

$$\Pr_{y \sim \{0,1\}^n}[B(y) = 1] \geq \epsilon$$

There exists $p \in S$ such that $B(p) = 1$.

$|\boldsymbol{S}| = \boldsymbol{n^{O(1)}}$ **solves** $\boldsymbol{RL} = \boldsymbol{L}$ (log-space computable S; $w = n$)

PRGs and Hitting Sets

- Fact: There exists a PRG with seed length $O(\log \frac{nw}{\epsilon})$
- [Nisan'92]: Explicit PRG with seed length

$$O(\log^2 n + \log n \log w + \log n \log 1/\epsilon)$$

- [Braverman-Cohen-Garg'18]: Explicit hitting set with seed length

$$\tilde{O}(\log^2 n + \log n \log w + \log 1/\epsilon)$$

Recent Improvements

[Braverman-Cohen-Garg'18]: Explicit pseudorandom pseudo-distribution with same seed length

[Hoza-Zuckerman'18]: Explicit hitting set with seed length

$$O(\log^2 n + \log n \log w + \log 1/\epsilon)$$

[Chattopadhyay-Liao'20]: Explicit pseudorandom pseudo-distribution with seed length

$$\tilde{O}(\log^2 n + \log n \log w) + O(\log 1/\epsilon)$$

Pseudorandom Pseudo-Distribution

A (w, n, ϵ)-Pseudo PRG is a sequence $(\rho_1, p_1), (\rho_2, p_2), \ldots, (\rho_{2^l}, p_{2^l}) \in \mathbb{R} \times \{0,1\}^n$ such that for every width w length n ROBP B,

$$\sum \rho_i B(p_i) \approx_\epsilon \Pr_{y \sim \{0,1\}^n}[B(y) = 1]$$

l is the seed length.

$\{p_i\}$ is a Hitting Set

Pseudorandom Pseudo-Distribution

A (w, n, ϵ)-Pseudo PRG is a sequence $(\rho_1, p_1), (\rho_2, p_2), \ldots, (\rho_{2^l}, p_{2^l}) \in \mathrm{R} \times \{0,1\}^n$ such that for every width w length n ROBP B,

$$\sum \rho_i B(p_i) \approx_\epsilon \Pr_{y \sim \{0,1\}^n}[B(y) = 1]$$

Pseudo-distribution approximates acceptance probability

Motivating Observation

[BCG'18, CL'20]: Explicit PRPD with seed length

$$\tilde{O}(\log^2 n + \log n \log w + \log 1/\epsilon)$$

[Saks-Zhou'99]: Any space s two-sided randomized algorithm can be simulated deterministically in space $O(s^{1.5})$. $\boldsymbol{BPL \subseteq L^{1.5}}$

Motivating Observation

[CL'20], [Saks-Zhou'99], [BCG'18]=> If one shows an explicit PRPD with seed length

$$O(\log^2 n + \log w + \log 1/\epsilon)$$

then $\boldsymbol{BPL \subseteq L^{4/3}}$

[Hoza-Zuckerman'18] Construction

Main Lemma: For every width w length n ROBP,

If $\Pr(v_1 \to 1) \geq \epsilon$, then there exists a layer and set of vertices $S(v_1)$ in that layer such that

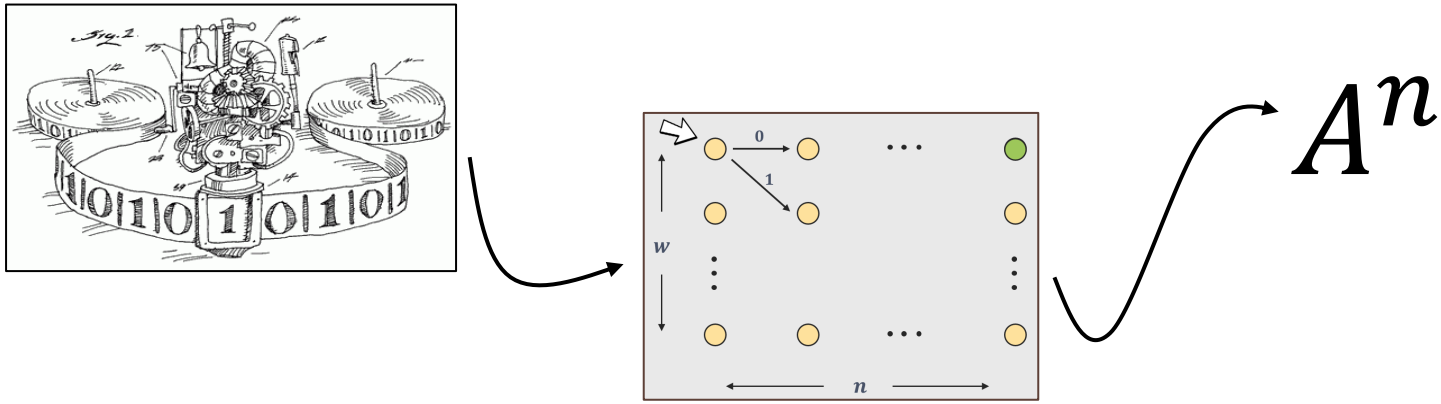
$$\Pr(v_1 \to S(v_1)) \geq \frac{2}{n^2}$$

$$\forall v \in S(v_1),\ \Pr(v \to 1) \geq \epsilon \frac{n}{4}$$

[Hoza-Zuckerman'18] Construction

- [Nisan'92] or [Arm'98]'s PRG with error parameter $1/n^2$ (seed length $O(\log^2 n)$) ⇒ reach $S(v_1)$ with probability at least $\frac{1}{n^2}$
- Use a hitter to hit a good PRG seed for every v_1 (extra $O(\log n)$ seed length)
- Recursing $O\left(\frac{\log 1/\epsilon}{\log n}\right)$ times is enough to hit a path that reaches accept state ($v_1 \to v_2 \to v_3 \to \cdots \to 1$, such that $v_{i+1} \in S(v_i)$)

Matrix-Product Perspective



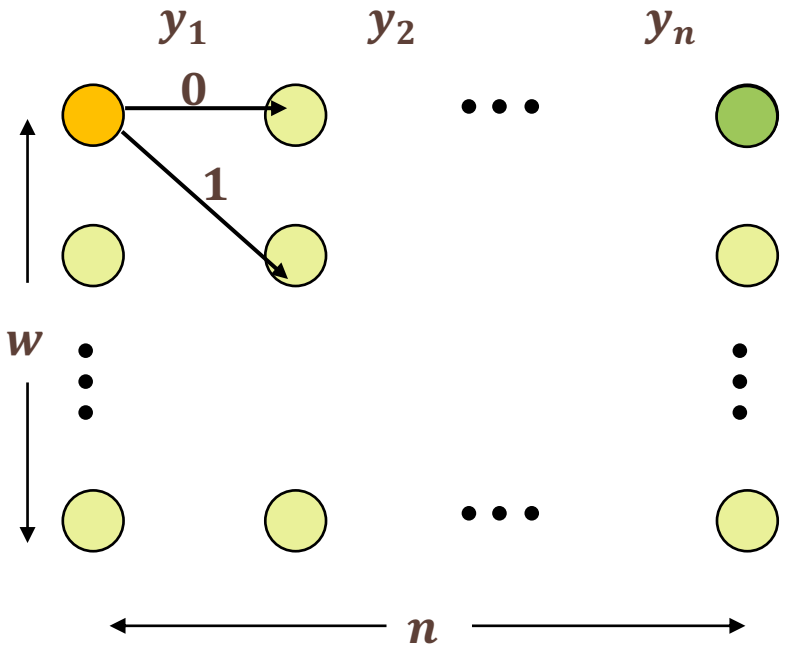
Matrix-Product Perspective

$$A_1 = \frac{1}{2}A_{10} + \frac{1}{2}A_{11}$$

A_{10} has exactly one 1 in each row;

A_1 is $w \times w$ stochastic matrix

Similarly, $A_2 = \frac{1}{2}A_{20} + \frac{1}{2}A_{21}$



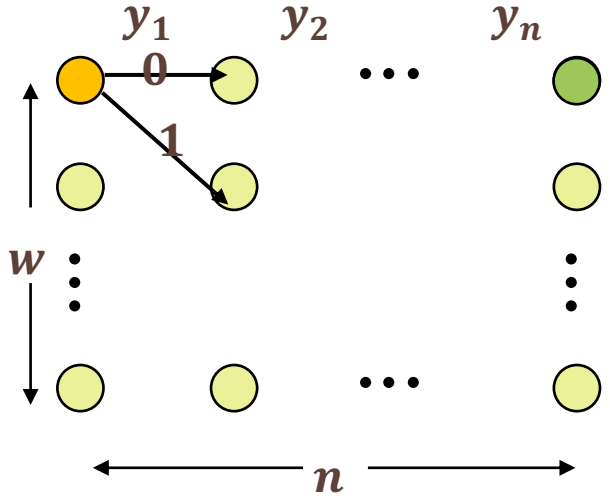
Matrix-Product Perspective

- $A_1A_2 = \frac{1}{4}A_{10}A_{20} + \frac{1}{4}A_{10}A_{21} + \frac{1}{4}A_{11}A_{20} + \frac{1}{4}A_{11}A_{21}$

$$= \frac{1}{4}\left(A_{\{00\}} + A_{\{01\}} + A_{\{10\}} + A_{\{11\}}\right)$$

- $A = A_1A_2 \dots A_n = \frac{1}{2^n}\sum_{y\in\{0,1\}^n} A_y$

Want to approximate $A_{(start,accept)}$



Approximating Matrix-Products

Given $w \times w$ stochastic matrices $A_1, A_2, \ldots, A_n$, entries $i, j \in [w]$, approximate $(A = A_1 A_2 \ldots A_n)_{i,j}$ up to an additive error ϵ.

Sparsifying Matrix-Product

$$A \approx_\varepsilon \frac{1}{2^r} \sum_{y \in \{0,1\}^r} A_y = \langle A \rangle$$

$$B \approx_\varepsilon \frac{1}{2^r} \sum_{y \in \{0,1\}^r} B_y = \langle B \rangle$$

Naive Product: $AB \approx_{2\varepsilon} \langle A \rangle \langle B \rangle = \frac{1}{2^{2r}} \sum_{y,t \in \{0,1\}^r} A_y B_t$

Sparsifying Matrix-Product

$$A \approx_\varepsilon \frac{1}{2^r} \sum_{y \in \{0,1\}^r} A_y = \langle A \rangle$$

$$B \approx_\varepsilon \frac{1}{2^r} \sum_{y \in \{0,1\}^r} B_y = \langle B \rangle$$

$$\langle A \circ_d B \rangle \approx_{2\epsilon+\delta} AB$$

Sparsified Product: $\langle A \circ_d B \rangle \approx_\delta \langle A \rangle \langle B \rangle = \frac{1}{2^{r+d}} \sum_{\substack{y \in \{0,1\}^r \\ t \in \{0,1\}^d}} A_y B_{f(y,t)}$

Dependence on error (Nisan's PRG)

- f can be defined using extractors, expanders, samplers, [Nis'92],[NZ'96],[RR'99]….
- $\langle A \circ_d B \rangle \approx_\delta \langle A \rangle \langle B \rangle$ for $d = \log(w/\delta)$
- As $\langle A \circ_d B \rangle \approx_{2\epsilon+\delta} AB$, error for multiplying n matrices is

$$\varepsilon(n) = 2\varepsilon(n/2) + \delta = \cdots = n\delta$$

Taking $\delta = \epsilon/n$, seed length

$$l(n) = l\left(\frac{n}{2}\right) + \log(wn/\epsilon) = \log n \cdot \log(wn/\epsilon)$$

The Idea [BCG’18]

The Delta Product

$\langle A \circ_k B \rangle$ (think of $k = \log \frac{wn}{\epsilon}$)

$$= \langle A \circ_g B \rangle + \langle A \circ_{2g} B \rangle - \langle A \circ_g B \rangle + \langle A \circ_{4g} B \rangle - \langle A \circ_{2g} B \rangle + \ldots \langle A \circ_k B \rangle - \langle A \circ_{k/2} B \rangle$$

$$= \langle A \circ_g B \rangle + \langle A \circ_{2g-g} B \rangle + \langle A \circ_{4g-g} B \rangle + \cdots \langle A \circ_{k-k/2} B \rangle$$

Properties of Delta Product

Linearity. $\langle A \circ_{D-d} B \rangle = \langle A \circ_D B \rangle - \langle A \circ_d B \rangle$

Smallness is stored: $||A \circ_{D-d} B|| \approx \frac{||A|| \cdot ||B||}{2^d}$

- $\langle A \circ_{D-d} B \rangle$ represents the difference of 2^{-D} and 2^{-d} approximation of AB, so it is small
- Further, the norm of the matrix product and the delta product scale with the matrix norms of A and B

Smallness is stored

Data Structure for Storing a Matrix

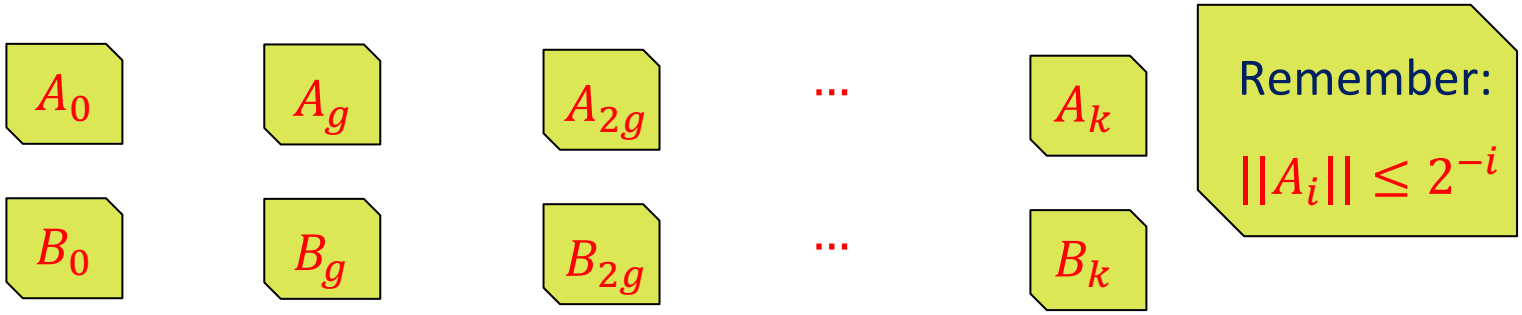
The data structure $\boldsymbol{A}$ consists of pieces $A_0, A_g, A_{2g}, A_{4g}, \ldots, A_k$ where $||A_i|| \leq 2^{-i}$.

$$\langle \boldsymbol{A} \rangle = \langle A_0 \rangle + \langle A_g \rangle + \langle A_{2g} \rangle + \cdots + \langle A_k \rangle$$

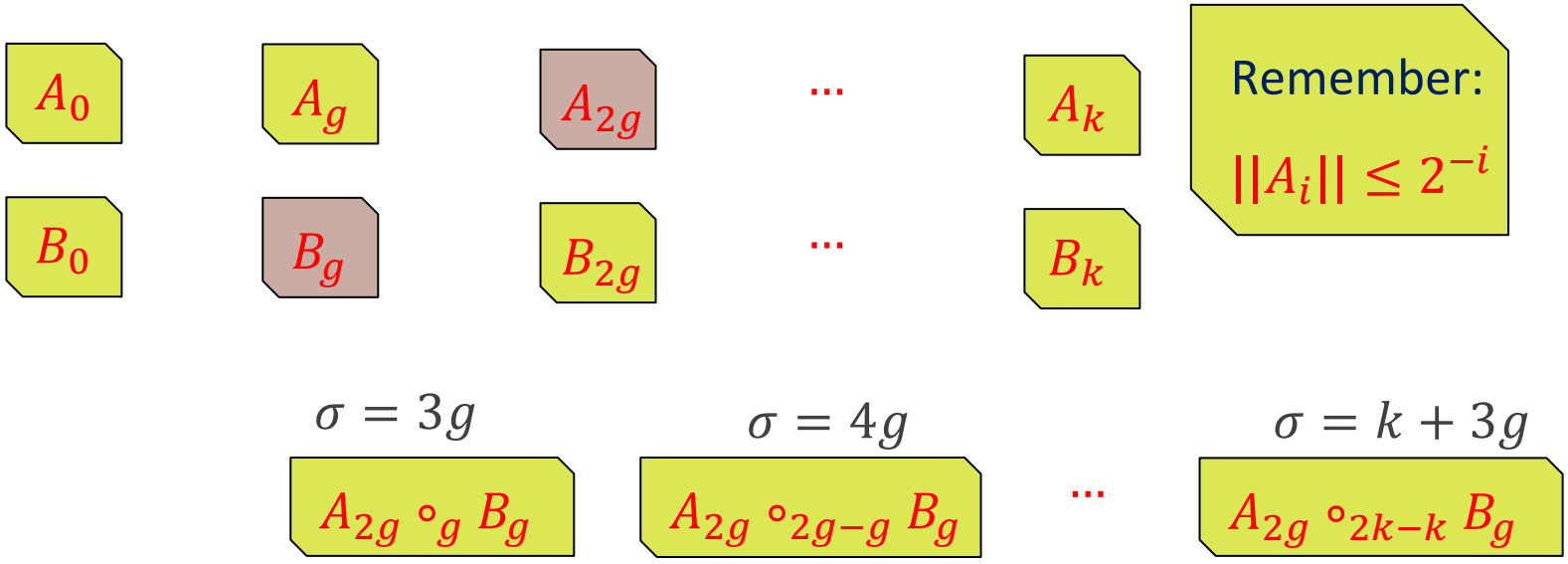
- Error terms that are small and give better approximations to the matrix product

$\langle \boldsymbol{A} \rangle \langle \boldsymbol{B} \rangle = \sum_{i,j} \langle A_i \rangle \langle B_j \rangle$. Next: define a product between pieces that approximates the product of error terms $\langle A_i \rangle \langle B_j \rangle$

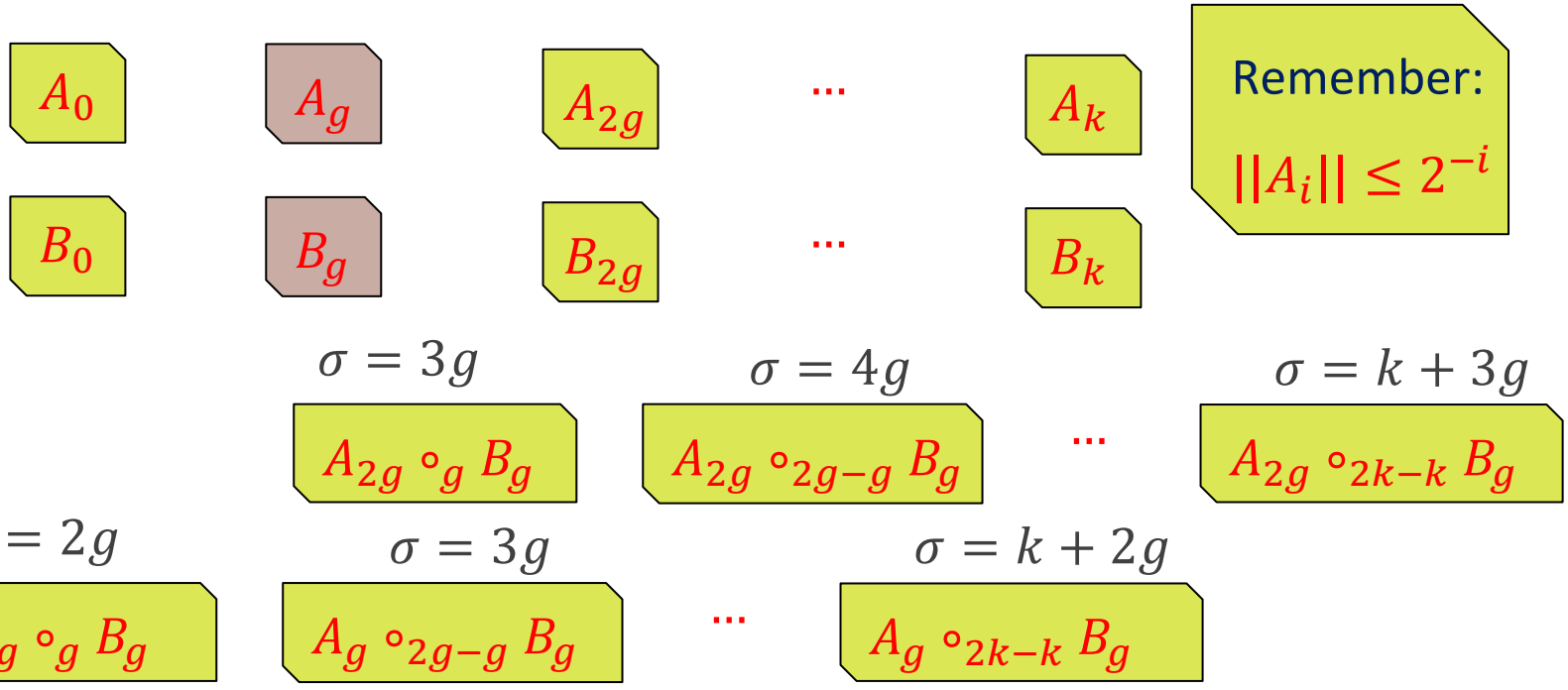
Multiplication for New Data Structure



Multiplication for New Data Structure



Multiplication for New Data Structure



Multiplication for New Data Structure

Add the terms with the same smallness for the data structure for the product of matrices

Chop off the error terms when they become smaller than ϵ

- Approximate with a low degree expander if error terms are already small
- As smallness is stored, any investment in seed length goes into the smallness. Therefore, almost optimal dependence on ϵ

A Bit More on the Construction

- Carefully defining the sparsified matrix products to ensure that small norm carries over to the product.
- Multiplying pieces of roughly the same smallness is done differently than multiplying pieces with significantly different smallness. This is where use of unbalanced samplers is crucial.

[Chattopadhyay-Liao'20]

Different approach for using rough approximations of a matrix

Main "top-down" Lemma: Let A_i be γ^{i+1}-approximation of A. Similarly for B. For $||A||, ||B|| \leq 1$,

$$\sum_{i=0}^{k} A_i B_{k-i} - \sum_{i=0}^{k-1} A_i B_{k-i-1}$$

is $(k+2)\gamma^{k+1} + (k+1)\gamma^{k+2}$- approximation of AB

References

[Nisan'92] Nisan, Noam. "Pseudorandom generators for space-bounded computation." *Combinatorica* 12.4 (1992): 449-461.

[NZ'96] Nisan, Noam, and David Zuckerman. "Randomness is linear in space." *Journal of Computer and System Sciences* 52.1 (1996): 43-52.

[Arm'98] Armoni, Roy. "On the derandomization of space-bounded computations." *International Workshop on Randomization and Approximation Techniques in Computer Science*. Springer, Berlin, Heidelberg, 1998.

References

[Saks-Zhou'99] M. Saks and S. Zhou. "BPHSPACE(s) ⊆ DSPACE(s^3/2)." Journal of computer and system sciences, 58(2):376–403, 1999.

[RR'99] Raz, Ran, and Omer Reingold. "On recycling the randomness of states in space bounded computation." *Proceedings of the thirty-first annual ACM symposium on Theory of computing*. 1999.

References

[Braverman-Cohen-Garg'18] Braverman, Mark, Gil Cohen, and Sumegha Garg. "Pseudorandom Pseudo-distributions with Near-Optimal Error for Read-Once Branching Programs." *SIAM Journal on Computing* 0 (2020): STOC18-242.

[Hoza-Zuckerman'18] Hoza, William, and David Zuckerman. "Simple optimal hitting sets for small-success RL." *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2018.

[Chattopadhyay-Liao'20] Chattopadhyay, Eshan, and Jyun-Jie Liao. "Optimal Error Pseudodistributions for Read-Once Branching Programs." *arXiv preprint arXiv:2002.07208* (2020).

Thank You!