HERMES

Efficient Ring Packing using MLWE Ciphertexts and Application to Transciphering

Youngjin Bae, Jung Hee Cheon, Jaehyung Kim,
Jai Hyun Park, Damien Stehlé

Summary

Summary

- We consider the ring packing problem.
  - Fully homomorphic encryption ring packing (FHE RP)

Summary

- We consider the ring packing problem.
  - Fully homomorphic encryption ring packing (FHE RP)
- We suggest generic acceleration tools for FHE RP.

Summary

- We consider the ring packing problem.
  - Fully homomorphic encryption ring packing (FHE RP)
- We suggest generic acceleration tools for FHE RP.
- We propose a new FHE RP method: HERMES.
  - 40x higher throughputs compared to state-of-the-art.

Summary

- We consider the ring packing problem.
  - Fully homomorphic encryption ring packing (FHE RP)
- We suggest generic acceleration tools for FHE RP.
- We propose a new FHE RP method: HERMES.
  - 40x higher throughputs compared to state-of-the-art.
- Application to transciphering.

Ring Packing

- Fully homomorphic encryption (FHE) is a cryptosystem that enables computations on encrypted data.

Ring Packing

- Fully homomorphic encryption (FHE) is a cryptosystem that enables computations on encrypted data.

- LWE format
  - Granularity and fast latency
  - TFHE / FHEW

Ring Packing

- Fully homomorphic encryption (FHE) is a cryptosystem that enables computations on encrypted data.

- LWE format
  - Granularity and fast latency
  - TFHE / FHEW

- Ring LWE format (RLWE)
  - Scalability and high throughput
  - BGV / BFV / CKKS

Ring Packing

- Fully homomorphic encryption (FHE) is a cryptosystem that enables computations on encrypted data.

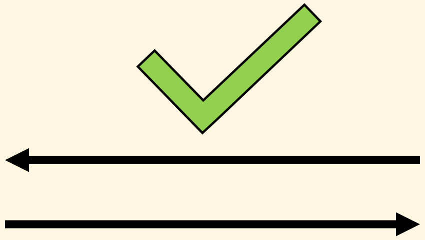
- LWE format
  - Granularity and fast latency
  - TFHE / FHEW

- Ring LWE format (RLWE)
  - Scalability and high throughput
  - BGV / BFV / CKKS

Ring Packing

- Fully homomorphic encryption (FHE) is a cryptosystem that enables computations on encrypted data.

- LWE format
  - Granularity and fast latency
  - TFHE / FHEW

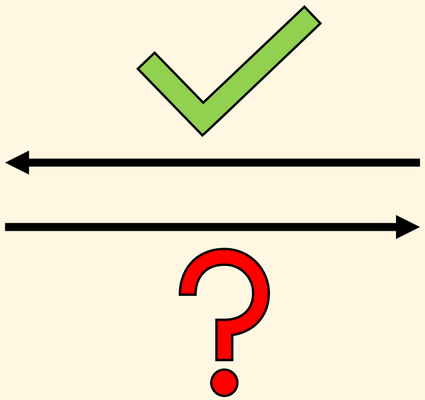


- Ring LWE format (RLWE)
  - Scalability and high throughput
  - BGV / BFV / CKKS

Ring Packing

- Fully homomorphic encryption (FHE) is a cryptosystem that enables computations on encrypted data.

- LWE format
  - Granularity and fast latency
  - TFHE / FHEW



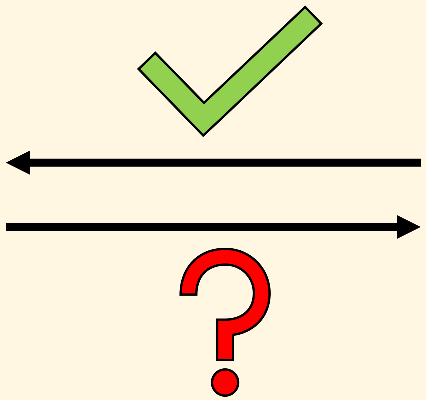
Ring packing

- Ring LWE format (RLWE)
  - Scalability and high throughput
  - BGV / BFV / CKKS

Ring Packing

- Fully homomorphic encryption (FHE) is a cryptosystem that enables computations on encrypted data.

- LWE format
  - Granularity and fast latency
  - TFHE / FHEW



Ring packing

- Ring LWE format (RLWE)
  - Scalability and high throughput
  - BGV / BFV / CKKS

- Ring packing (RP) bridges LWE and RLWE formats [CGGI17, MS18, BGGJ20, CDKS21, LHH+21]
  - Scheme switching during homomorphic computation
  - Transciphering

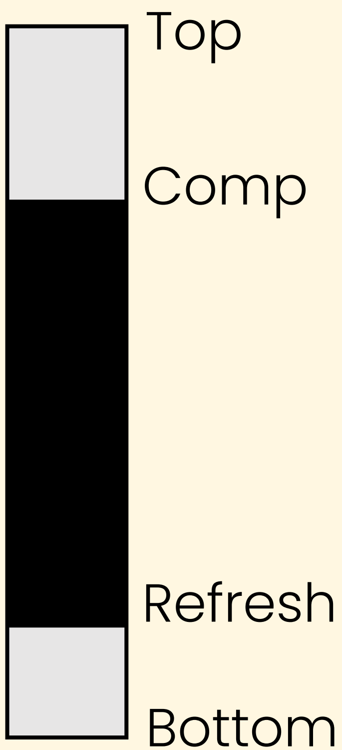
RLWE formats for FHE

RLWE formats for FHE

- RLWE-based FHE schemes
  - RLWE schemes are *leveled homomorphic encryptions.*
  - Parameters: Moduli and Ring degree
  - Encoding: Slots-encoding / Coefficients-encoding

RLWE formats for FHE

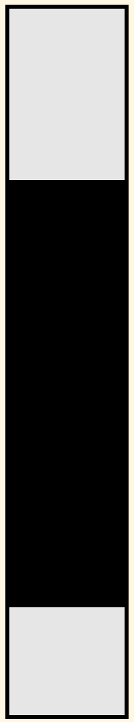
- RLWE-based FHE schemes
  - RLWE schemes are *leveled homomorphic encryptions.*
  - Parameters: Moduli and Ring degree
  - Encoding: Slots-encoding / Coefficients-encoding



RLWE formats for FHE

- RLWE-based FHE schemes
  - RLWE schemes are *leveled homomorphic encryptions.*
  - Parameters: Moduli and Ring degree
  - Encoding: Slots-encoding / Coefficients-encoding

- Which ring packing?

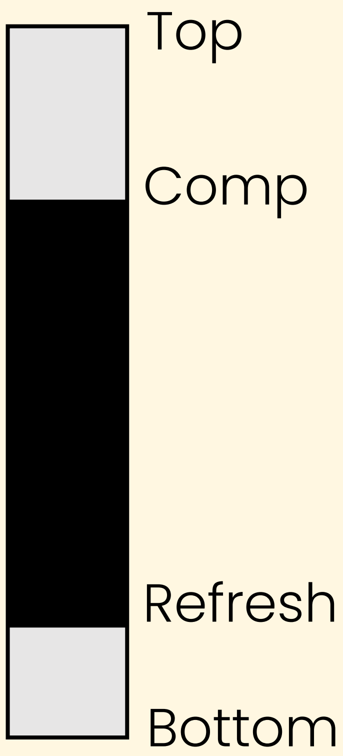


RLWE formats for FHE

- RLWE-based FHE schemes
  - RLWE schemes are *leveled homomorphic encryptions.*
  - Parameters: Moduli and Ring degree
  - Encoding: Slots-encoding / Coefficients-encoding

- Which ring packing?

- FHE Ring Packing (FHE RP)



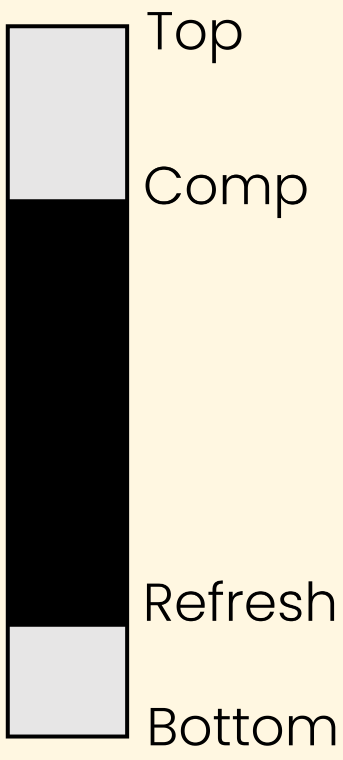
RLWE formats for FHE

- RLWE-based FHE schemes
  - RLWE schemes are *leveled homomorphic encryptions.*
  - Parameters: Moduli and Ring degree
  - Encoding: Slots-encoding / Coefficients-encoding

- Which ring packing?

- FHE Ring Packing (FHE RP)

"Packing into slots-encoding RLWE of modulus Q_{comp} and degree N_{BTS}."



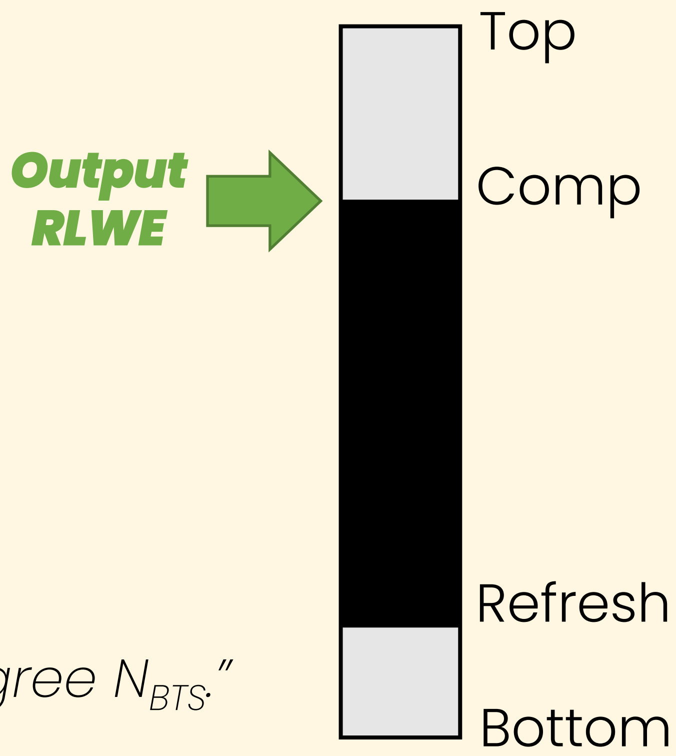
RLWE formats for FHE

- RLWE-based FHE schemes
  - RLWE schemes are *leveled homomorphic encryptions.*
  - Parameters: Moduli and Ring degree
  - Encoding: Slots-encoding / Coefficients-encoding

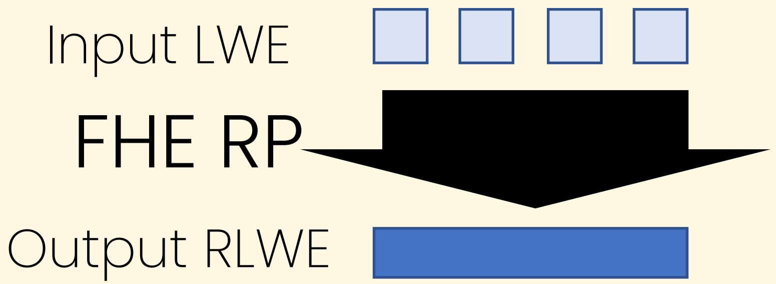
- Which ring packing?

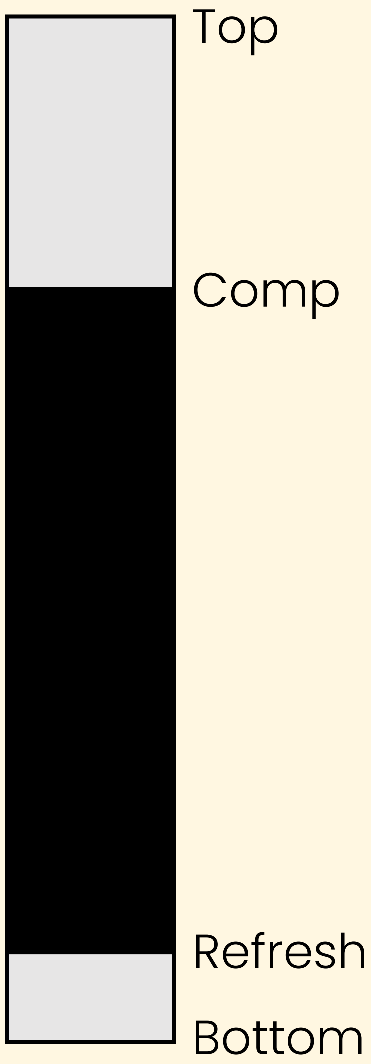
- FHE Ring Packing (FHE RP)

"Packing into slots-encoding RLWE of modulus Q_{comp} and degree N_{BTS}."



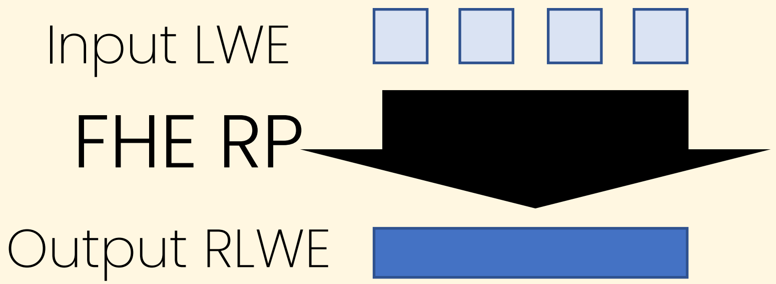
Large parameters for FHE RP

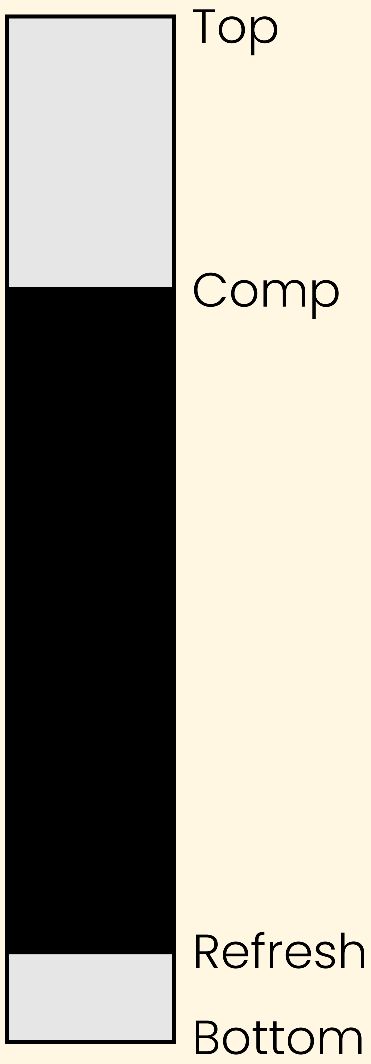




- FHE RP means outputting RLWE with large parameters.

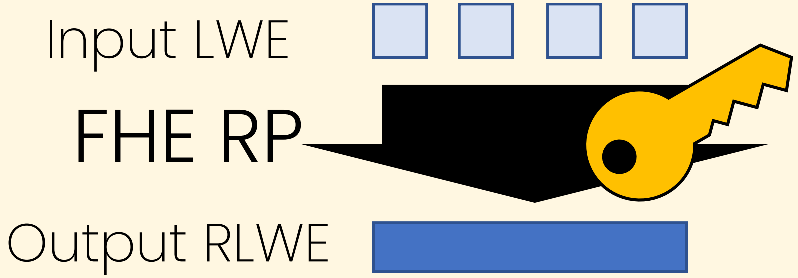
Large parameters for FHE RP

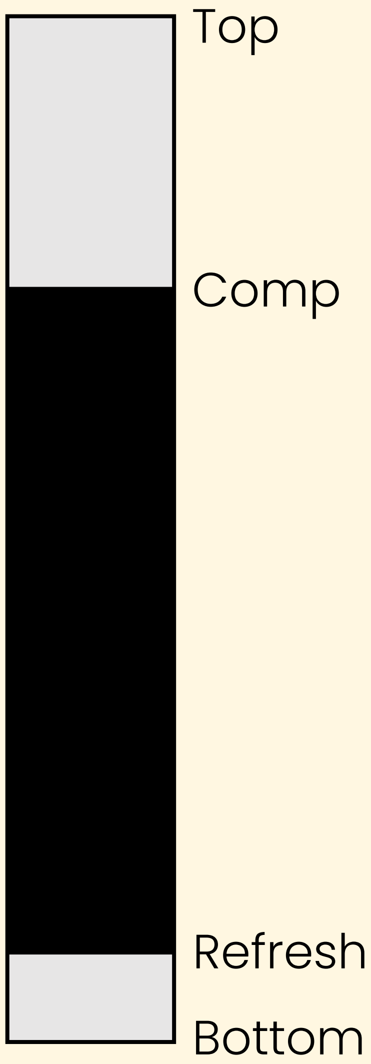




- FHE RP means outputting RLWE with large parameters.
- Unsatisfactory runtime and key size.
  - Computation in higher moduli and degree is slow.

Large parameters for FHE RP



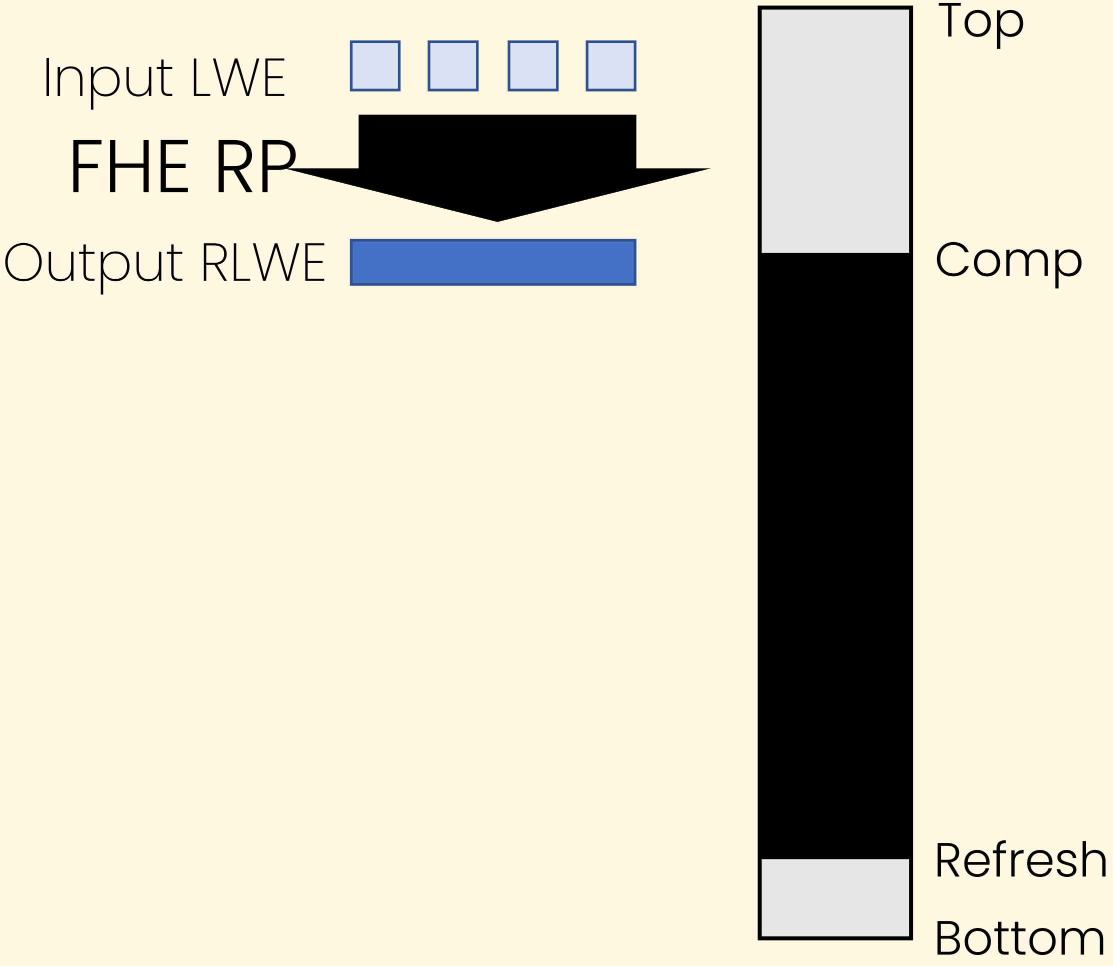


- FHE RP means outputting RLWE with large parameters.
- Unsatisfactory runtime and key size.
  - Computation in higher moduli and degree is slow.
  - Requires large evaluation keys.

Accelerating FHE RP

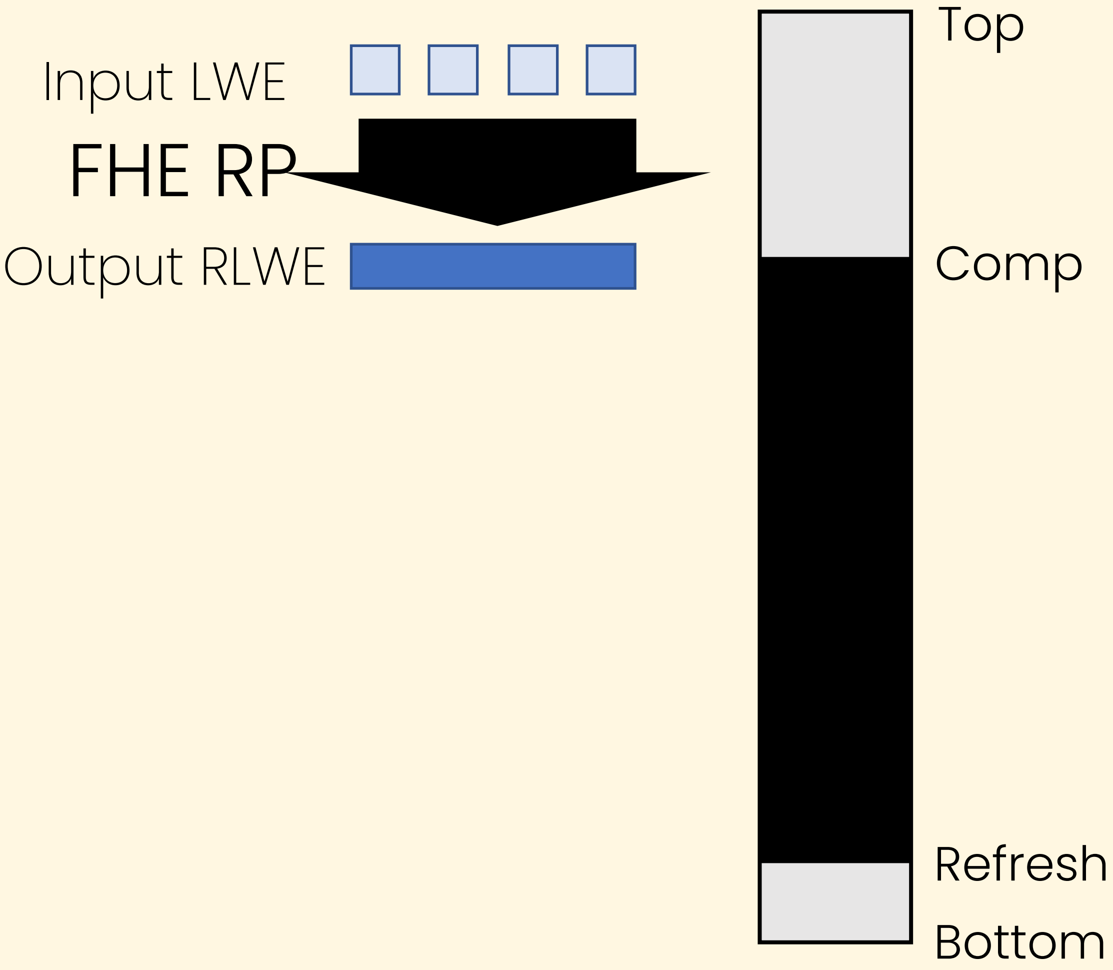
RLWE Moduli Optimization

Conventional approach

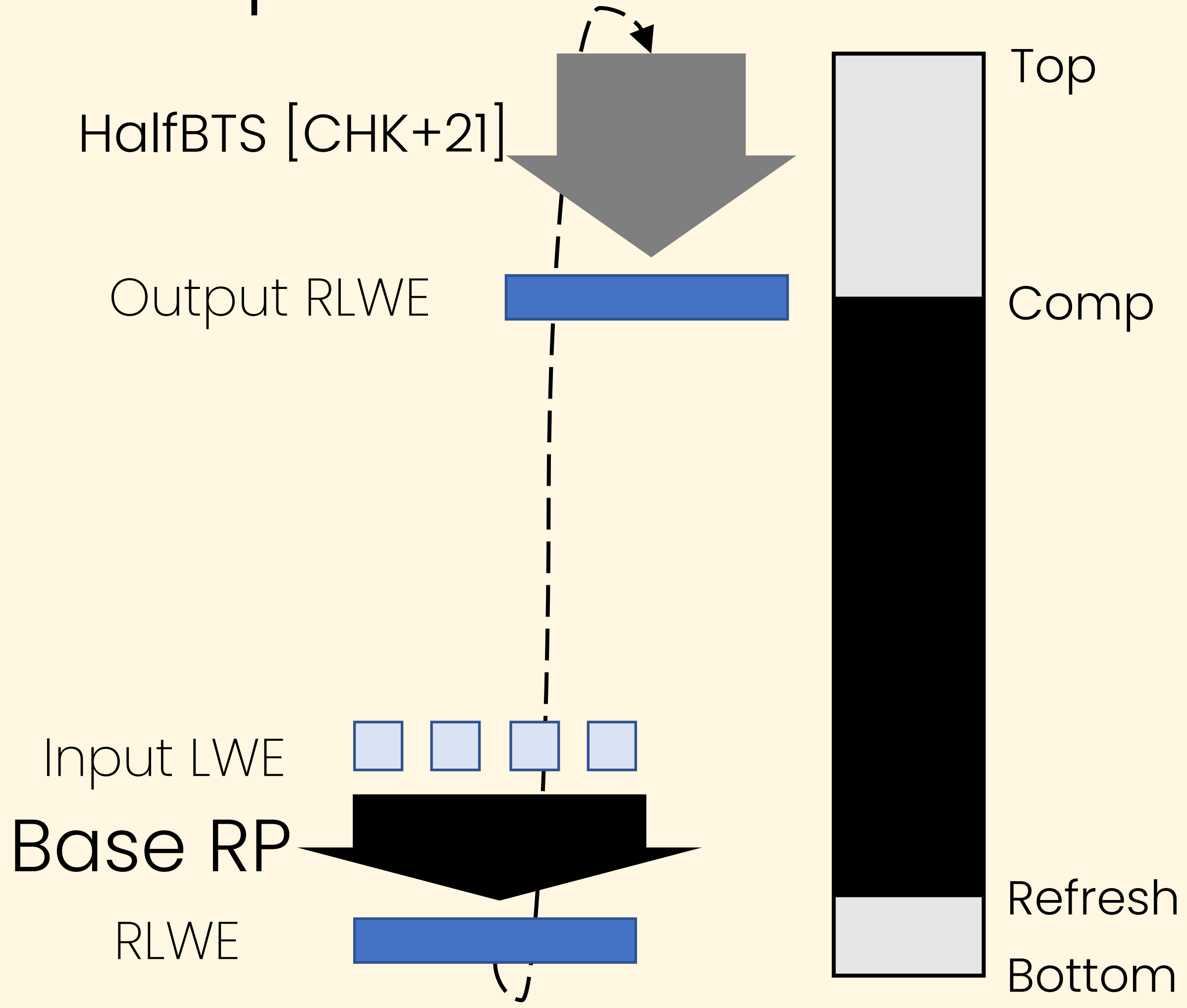


RLWE Moduli Optimization

Conventional approach

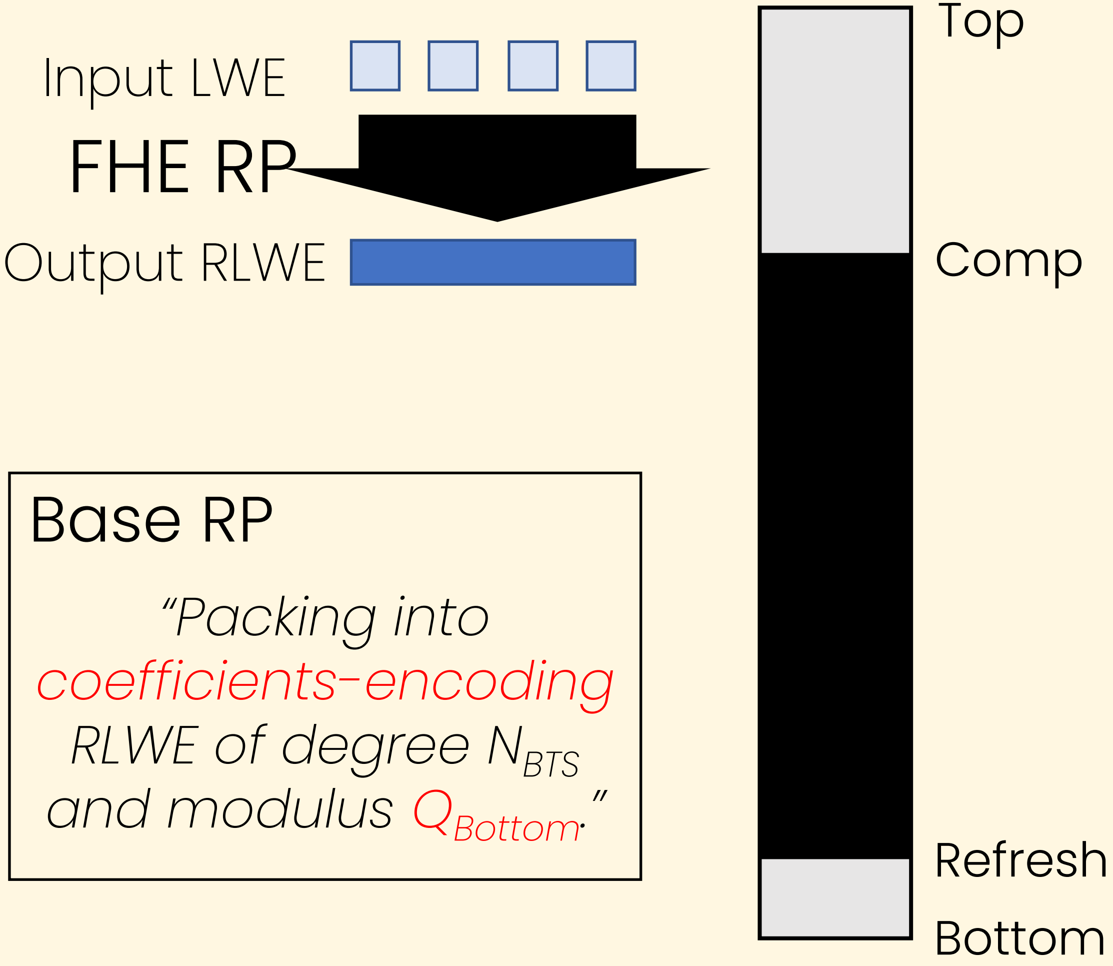


Moduli optimization



RLWE Moduli Optimization

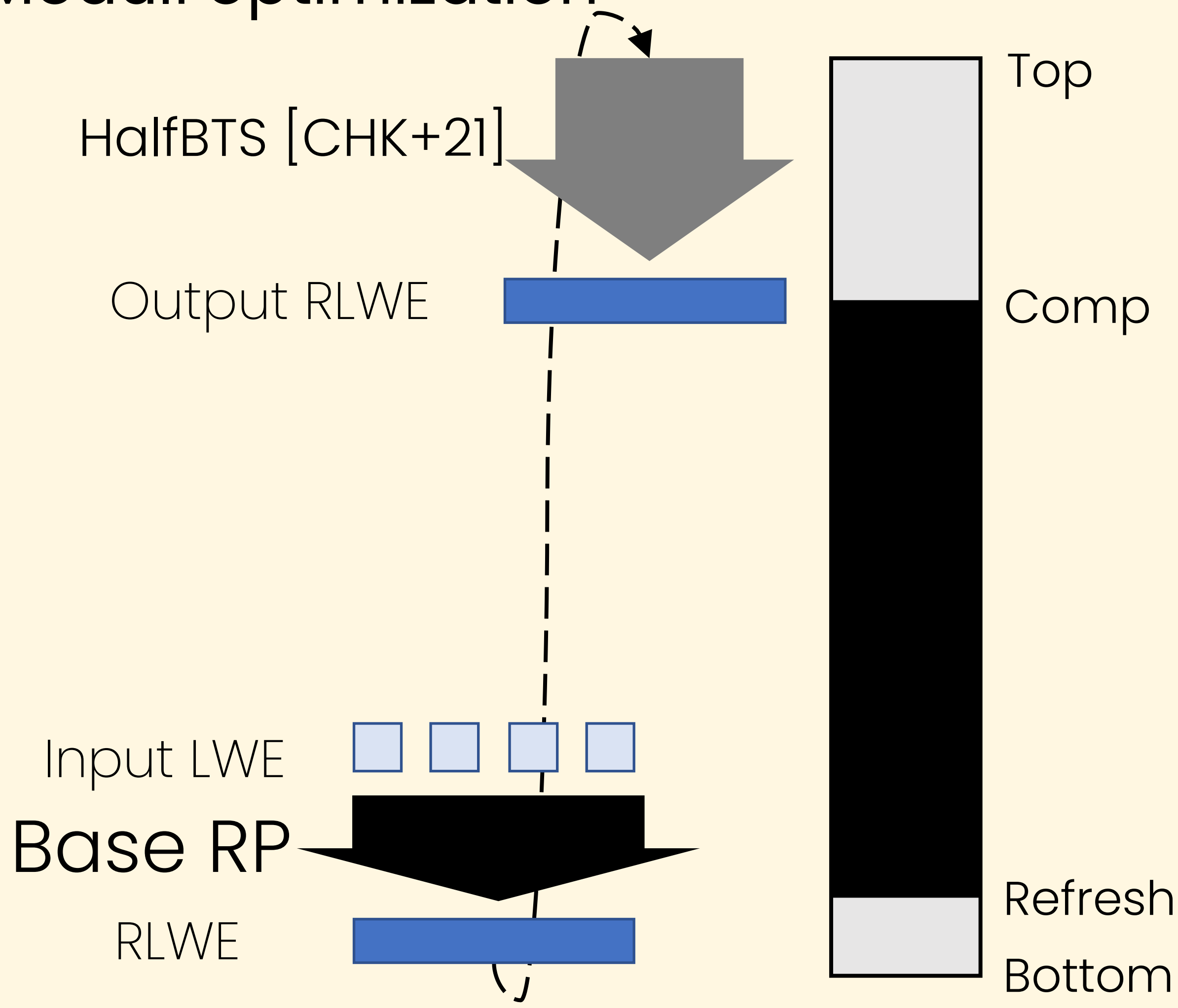
Conventional approach



Base RP

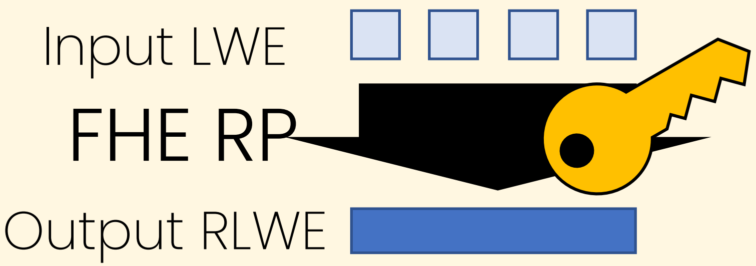
"Packing into coefficients-encoding RLWE of degree N_{BTS} and modulus Q_{Bottom}."

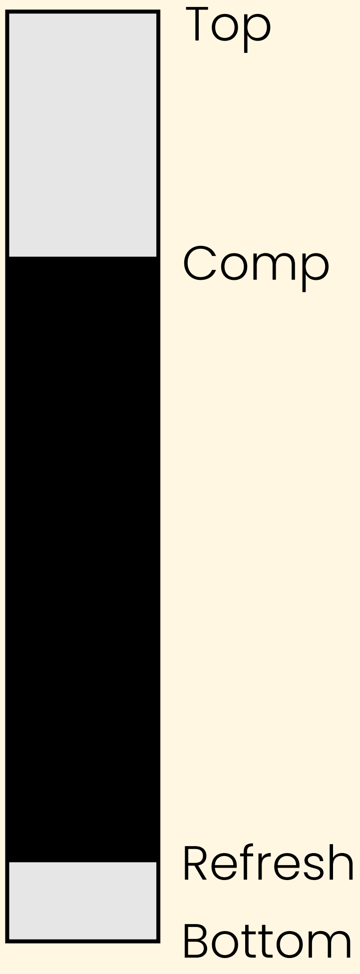
Moduli optimization



RLWE Moduli Optimization

Conventional approach

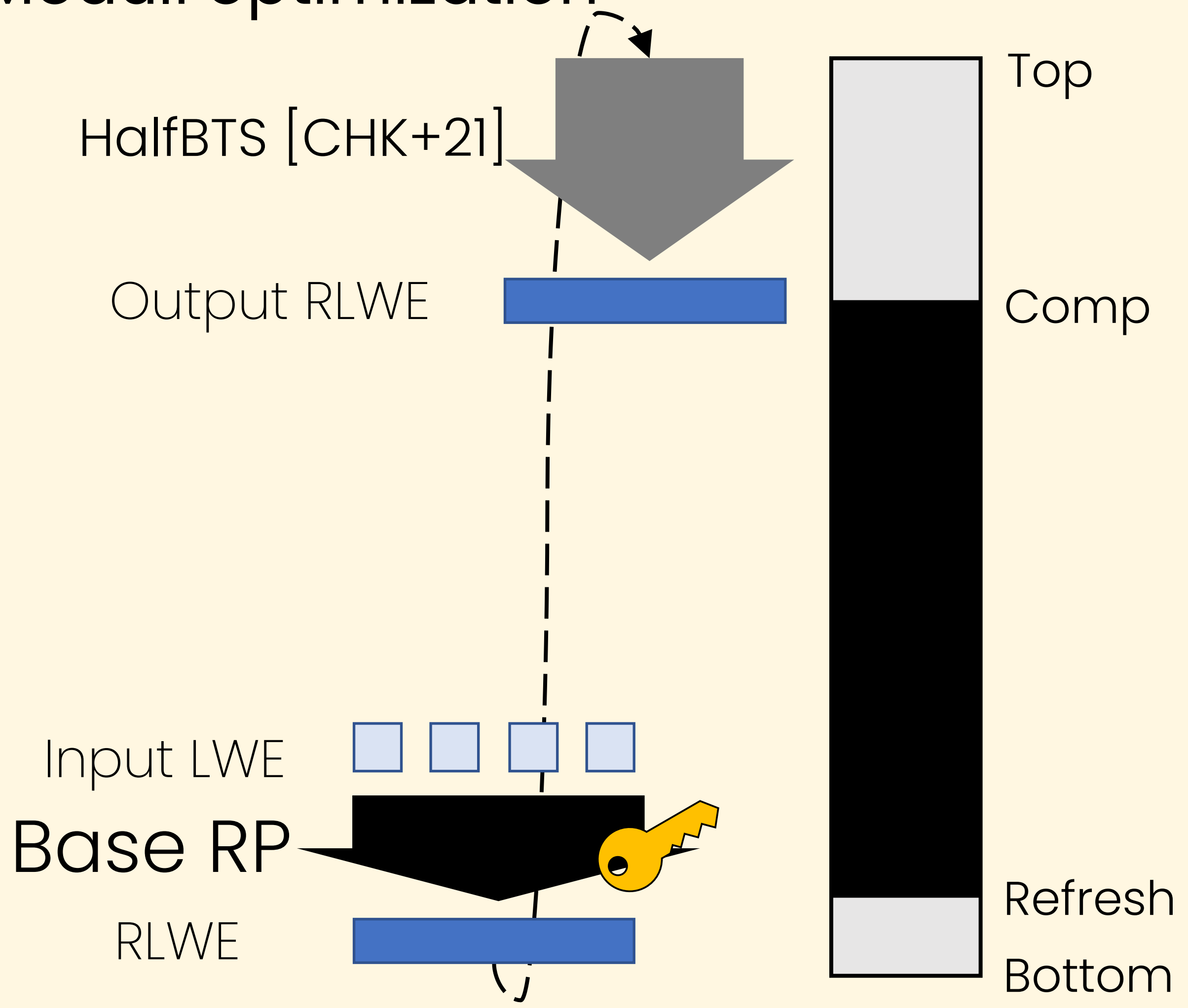




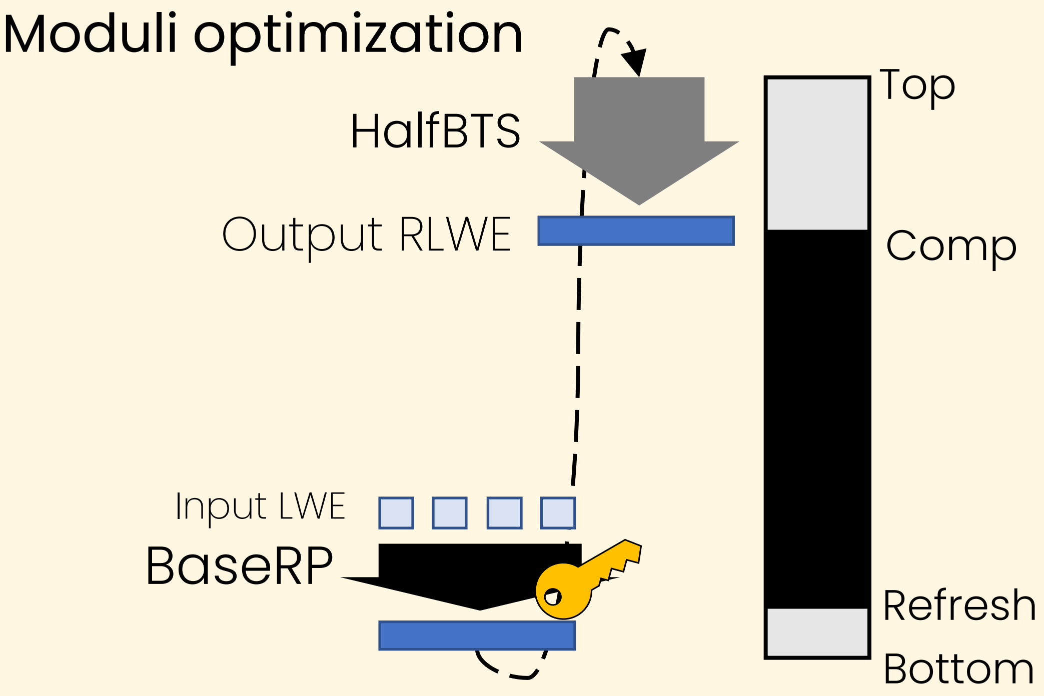
Base RP

"Packing into coefficients-encoding RLWE of degree N_{BTS} and modulus Q_{Bottom}."

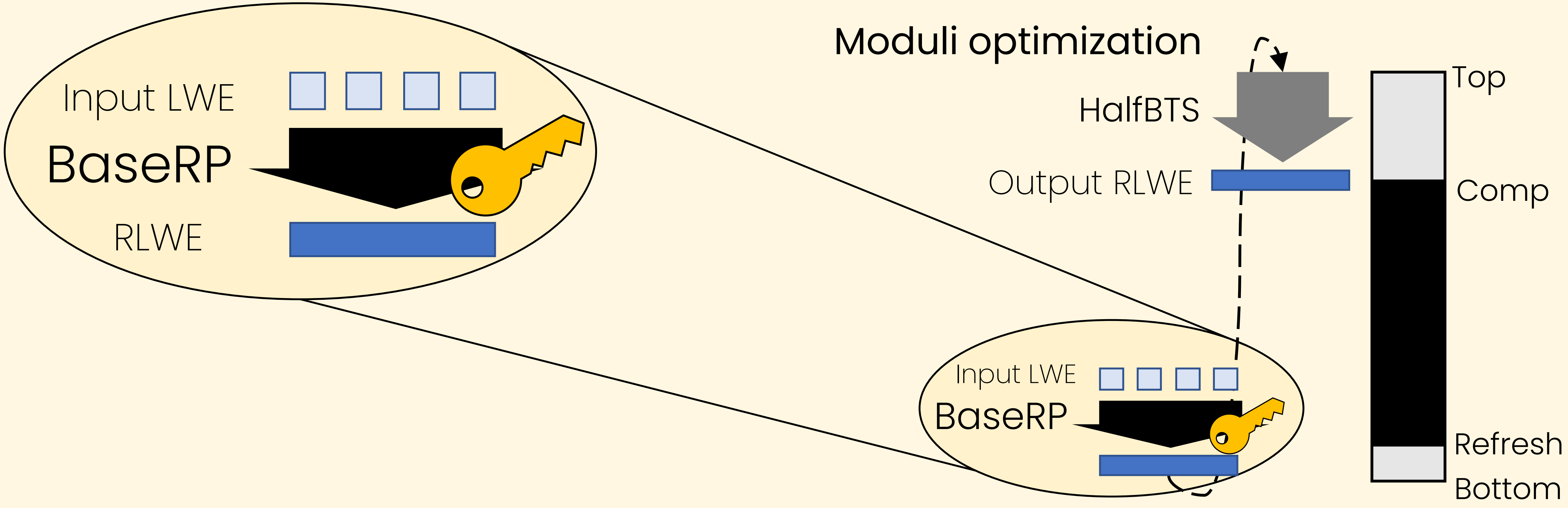
Moduli optimization



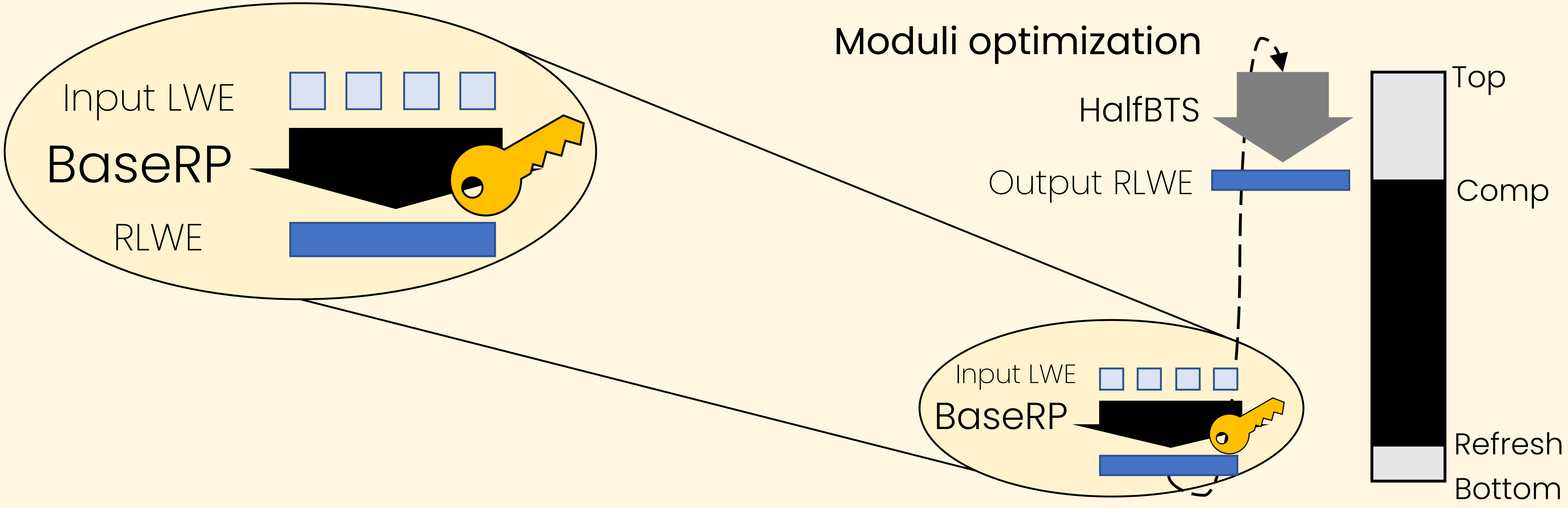
RLWE Degree Optimization



RLWE Degree Optimization



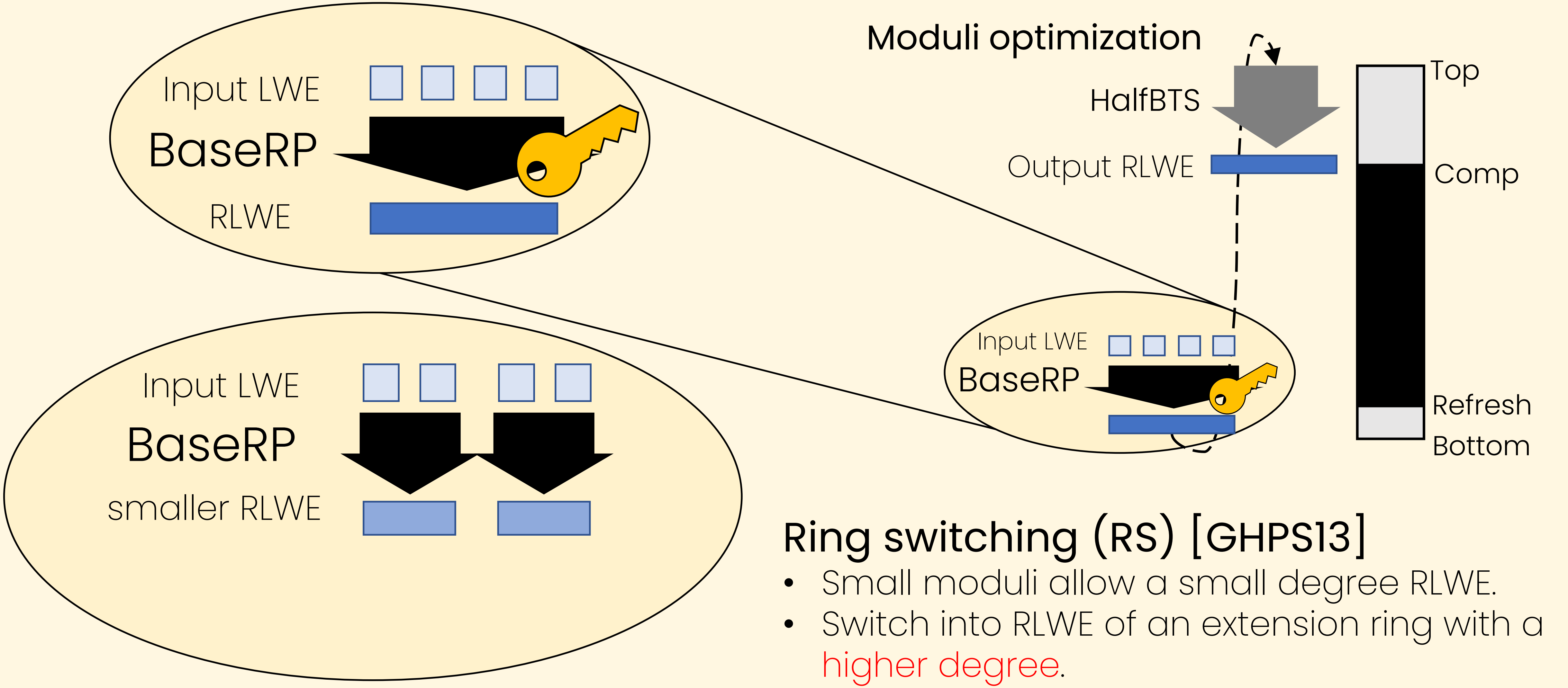
RLWE Degree Optimization



Ring switching (RS) [GHPS13]

- Small moduli allow a small degree RLWE.
- Switch into RLWE of an extension ring with a higher degree.

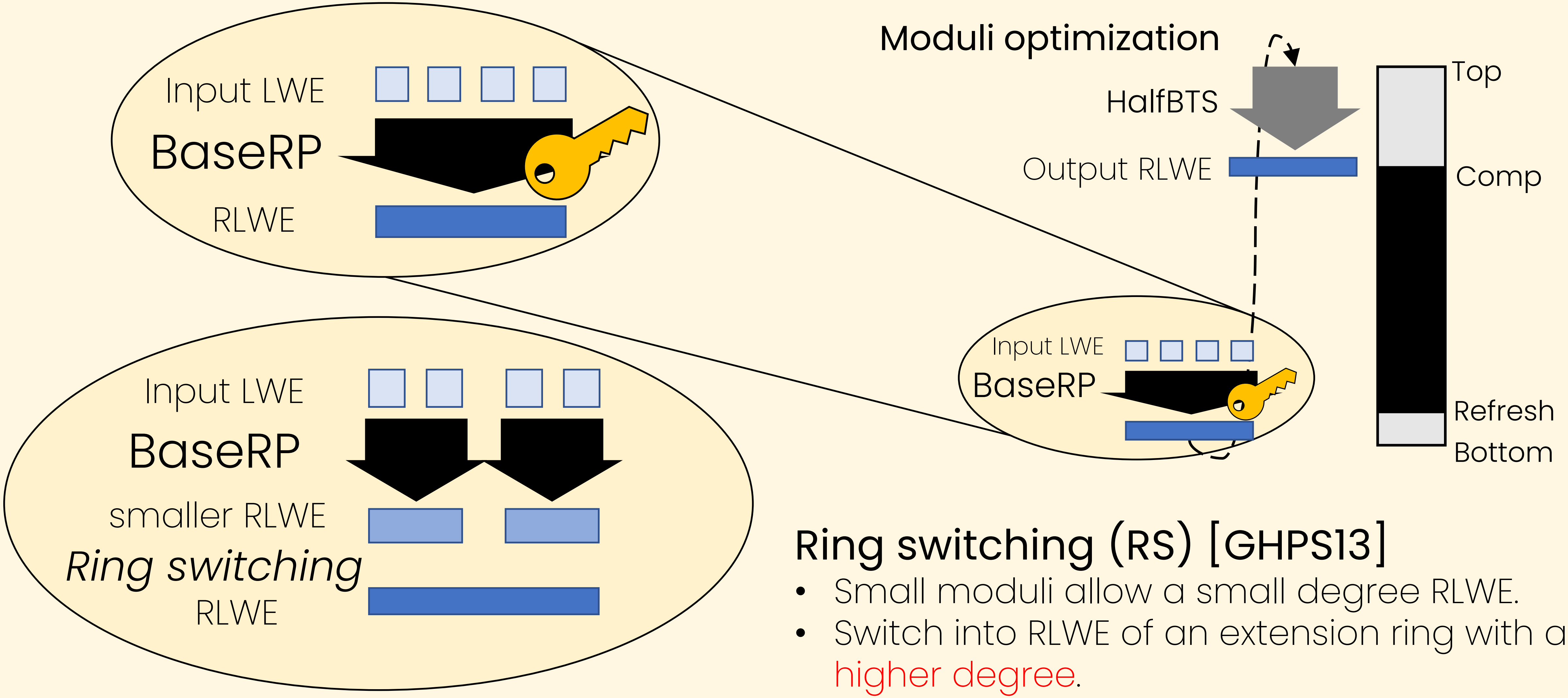
RLWE Degree Optimization



Ring switching (RS) [GHPS13]

- Small moduli allow a small degree RLWE.
- Switch into RLWE of an extension ring with a higher degree.

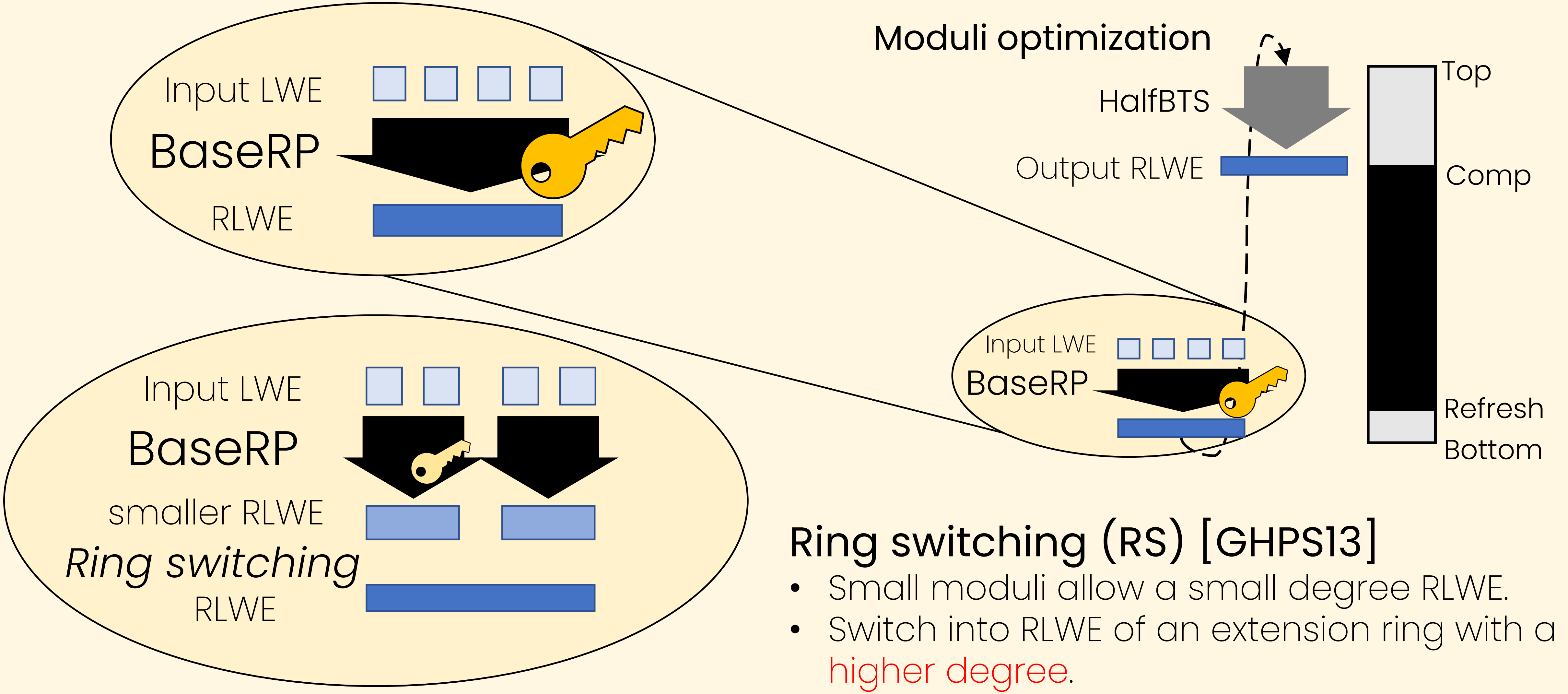
RLWE Degree Optimization



Ring switching (RS) [GHPS13]

- Small moduli allow a small degree RLWE.
- Switch into RLWE of an extension ring with a higher degree.

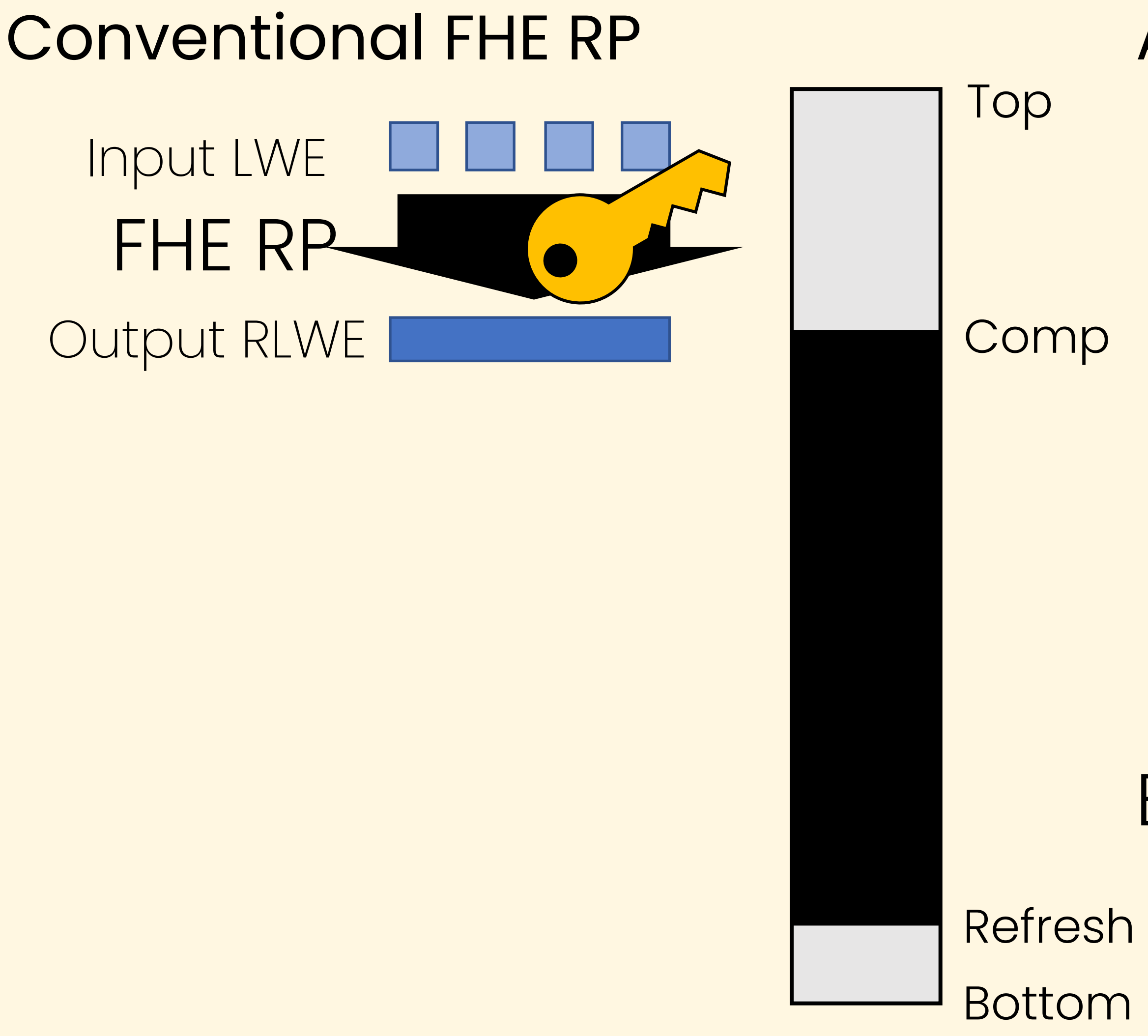
RLWE Degree Optimization

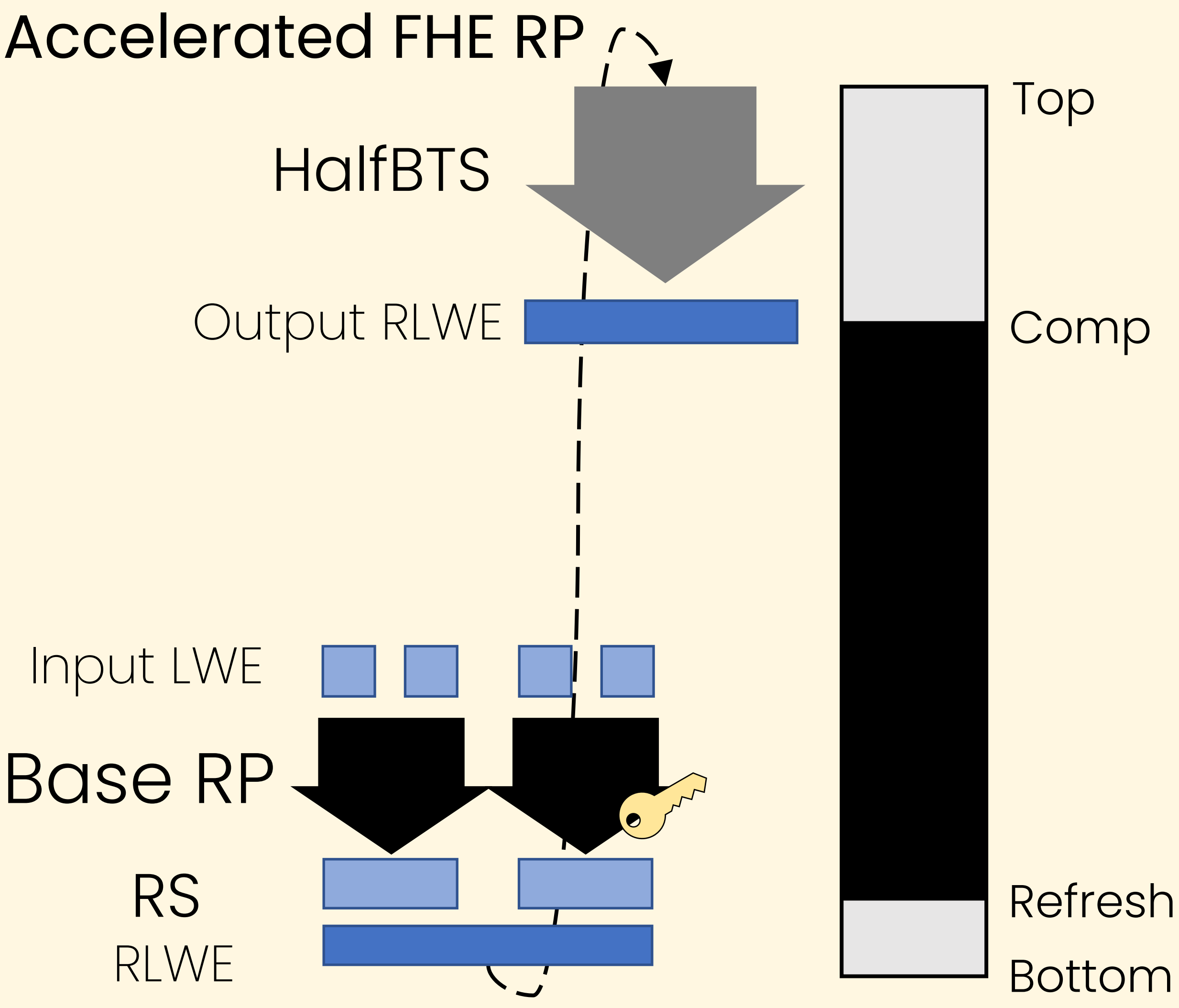


Ring switching (RS) [GHPS13]

- Small moduli allow a small degree RLWE.
- Switch into RLWE of an extension ring with a higher degree.

Improved FHE RP





Existing Approaches

RP as a Matrix Multiplication

- LWE ciphertexts $\mathbf{ct}_i$: $c_{i\,1}\,s_1 + c_{i\,2}\,s_2 + \cdots + c_{i\,K}\,s_K \approx m_i$ for each i

$$\begin{bmatrix} c_{1\,1} & c_{1\,2} & \cdots & c_{1\,K} \\ c_{2\,1} & c_{2\,2} & \cdots & c_{2\,K} \\ \vdots & \vdots & & \vdots \\ c_{N\,1} & c_{N\,2} & \cdots & c_{N\,K} \end{bmatrix} \times \begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_K \end{bmatrix} \approx \begin{bmatrix} m_1 \\ m_2 \\ \vdots \\ m_N \end{bmatrix}$$

RP as a Matrix Multiplication

- LWE ciphertexts $\mathbf{ct}_i$: $c_{i\,1}\, s_1 + c_{i\,2}\, s_2 + \cdots + c_{i\,K}\, s_K \approx m_i$ for each i

$$\begin{bmatrix} c_{1\,1} & c_{1\,2} & \cdots & c_{1\,K} \\ c_{2\,1} & c_{2\,2} & \cdots & c_{2\,K} \\ \vdots & \vdots & & \vdots \\ c_{N\,1} & c_{N\,2} & \cdots & c_{N\,K} \end{bmatrix} \times \begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_K \end{bmatrix} \approx \begin{bmatrix} m_1 \\ m_2 \\ \vdots \\ m_N \end{bmatrix}$$

… is a linear system with errors.

RP as a Matrix Multiplication

- LWE ciphertexts $\mathbf{ct}_i$: $c_{i\,1}\, s_1 + c_{i\,2}\, s_2 + \cdots + c_{i\,K}\, s_K \approx m_i$ for each $\boldsymbol{i}$

$$\begin{bmatrix} c_{1\,1} & c_{1\,2} & \cdots & c_{1\,K} \\ c_{2\,1} & c_{2\,2} & \cdots & c_{2\,K} \\ \vdots & \vdots & & \vdots \\ c_{N\,1} & c_{N\,2} & \cdots & c_{N\,K} \end{bmatrix} \times \boxed{\begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_K \end{bmatrix}} \approx \begin{bmatrix} m_1 \\ m_2 \\ \vdots \\ m_N \end{bmatrix}$$

… is a linear system with errors.

RP as a Matrix Multiplication

- LWE ciphertexts $\mathbf{ct}_i$: $c_{i\,1}\, s_1 + c_{i\,2}\, s_2 + \cdots + c_{i\,K}\, s_K \approx m_i$ for each i

$$\begin{bmatrix} c_{1\,1} & c_{1\,2} & \cdots & c_{1\,K} \\ c_{2\,1} & c_{2\,2} & \cdots & c_{2\,K} \\ \vdots & \vdots & & \vdots \\ c_{N\,1} & c_{N\,2} & \cdots & c_{N\,K} \end{bmatrix} \times \boxed{\begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_K \end{bmatrix}} \approx \boxed{\begin{bmatrix} m_1 \\ m_2 \\ \vdots \\ m_N \end{bmatrix}}$$

… is a linear system with errors.

RP as a Matrix Multiplication

- LWE ciphertexts $\mathbf{ct}_i$: $c_{i\,1}\, s_1 + c_{i\,2}\, s_2 + \cdots + c_{i\,K}\, s_K \approx m_i$ for each $\boldsymbol{i}$

$$\begin{bmatrix} c_{1\,1} & c_{1\,2} & \cdots & c_{1\,K} \\ c_{2\,1} & c_{2\,2} & \cdots & c_{2\,K} \\ \vdots & \vdots & & \vdots \\ c_{N\,1} & c_{N\,2} & \cdots & c_{N\,K} \end{bmatrix} \times \begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_K \end{bmatrix} \approx \begin{bmatrix} m_1 \\ m_2 \\ \vdots \\ m_N \end{bmatrix}$$

… is a linear system with errors.

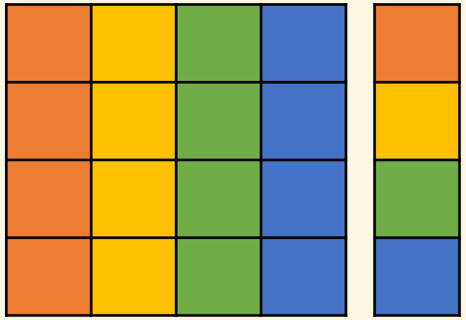
- RP is (plaintext) matrix – (ciphertext) vector multiplication in RLWE formats.

Existing Approaches

- Three approaches to encode the plaintext matrix.

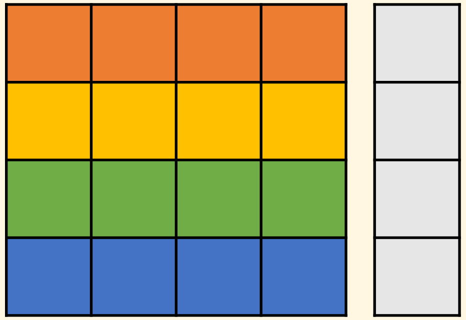
Column method

- CGGI17, BGGJ20
- K key switchings
- K keys
- Consumes $\mathbf{0}$ level



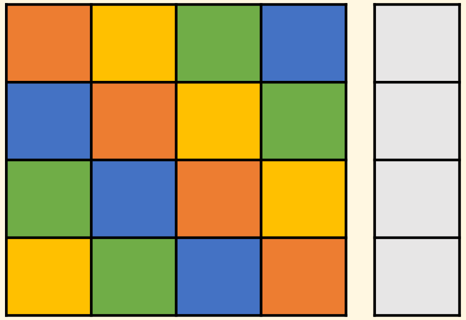
Row method

- CDKS21
- K key switchings
- $\log K$ keys
- Consumes $\mathbf{1}$ level

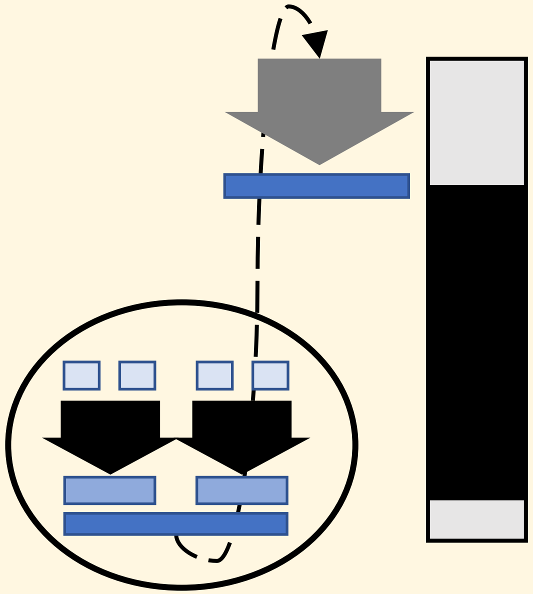


Diagonal method

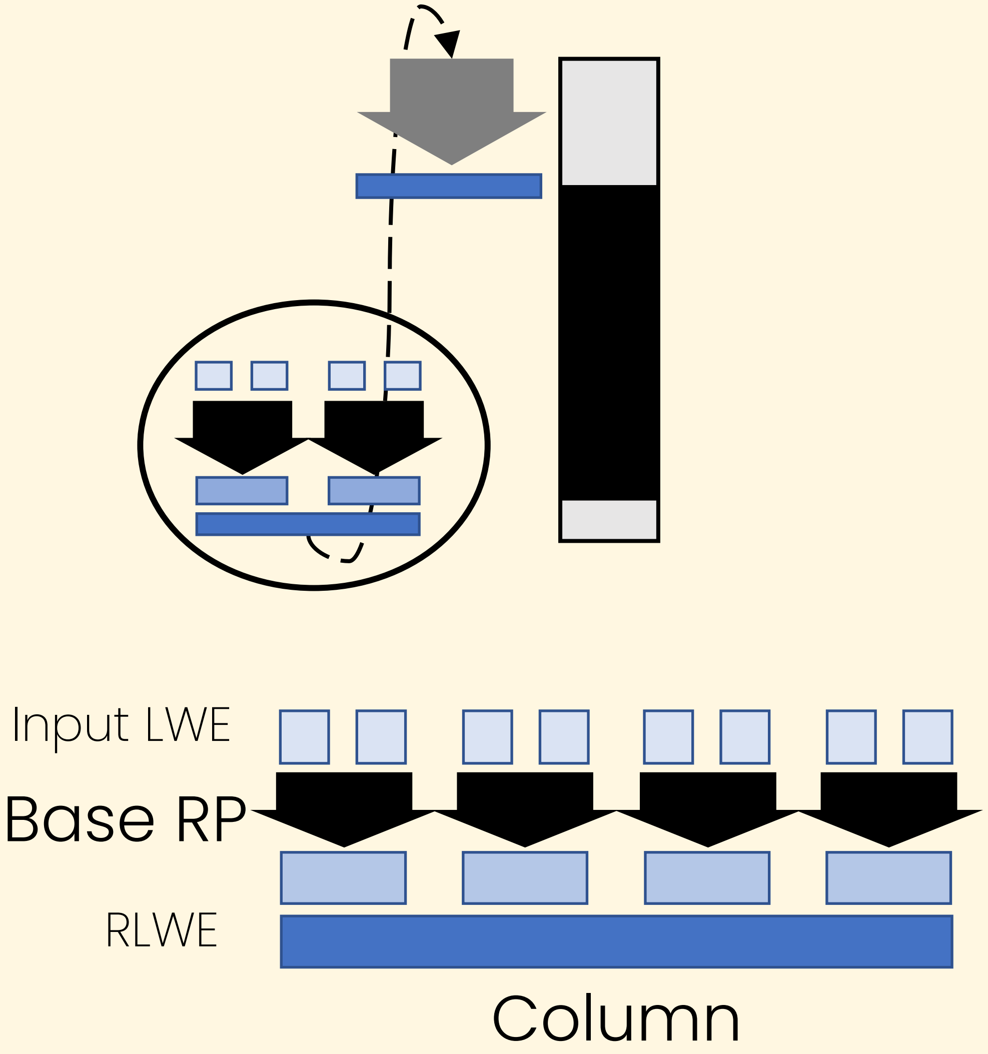
- HS14, LHH+21
- $2\sqrt{K}$ key switchings
- $2\sqrt{K}$ keys
- Consumes ≥ 4 levels



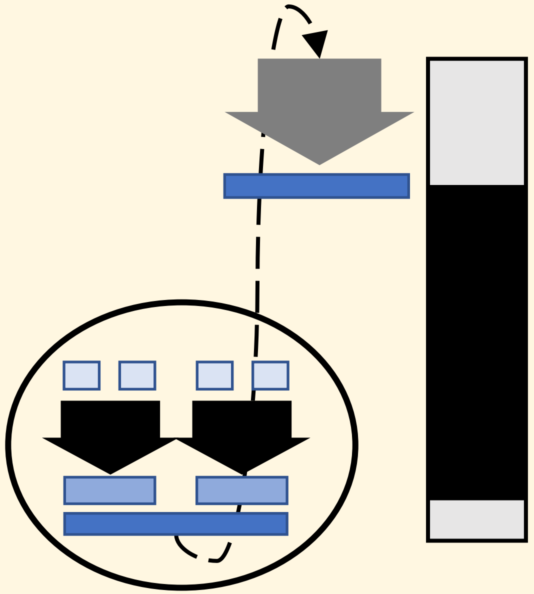
Base RP with Existing Approaches

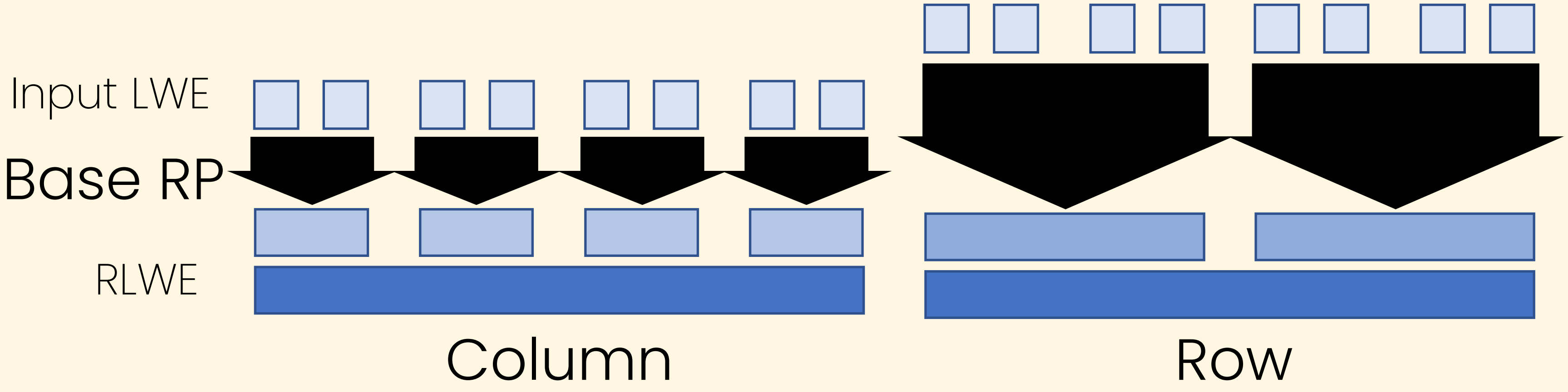


Base RP with Existing Approaches

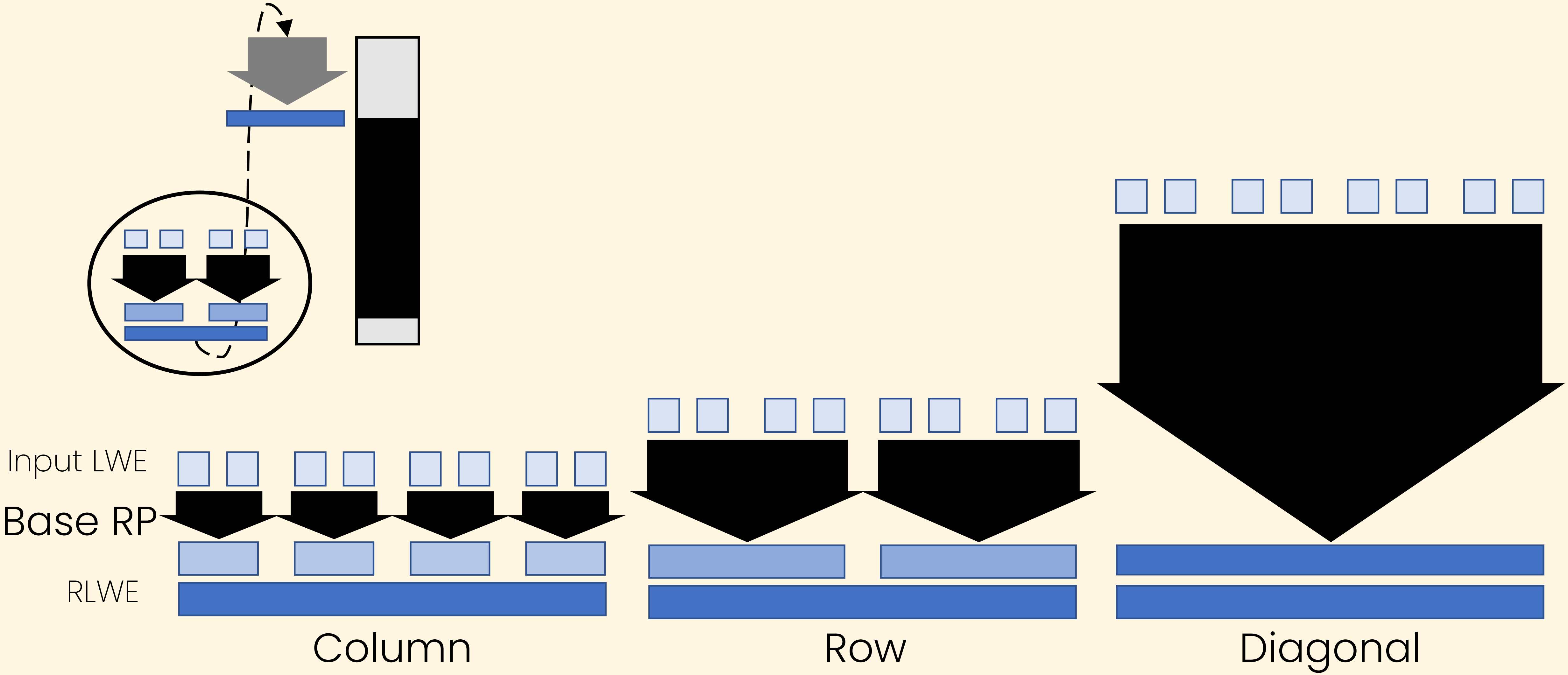


Base RP with Existing Approaches

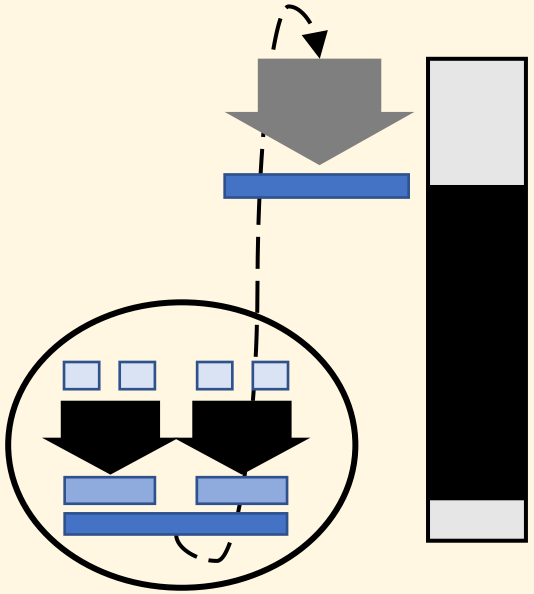




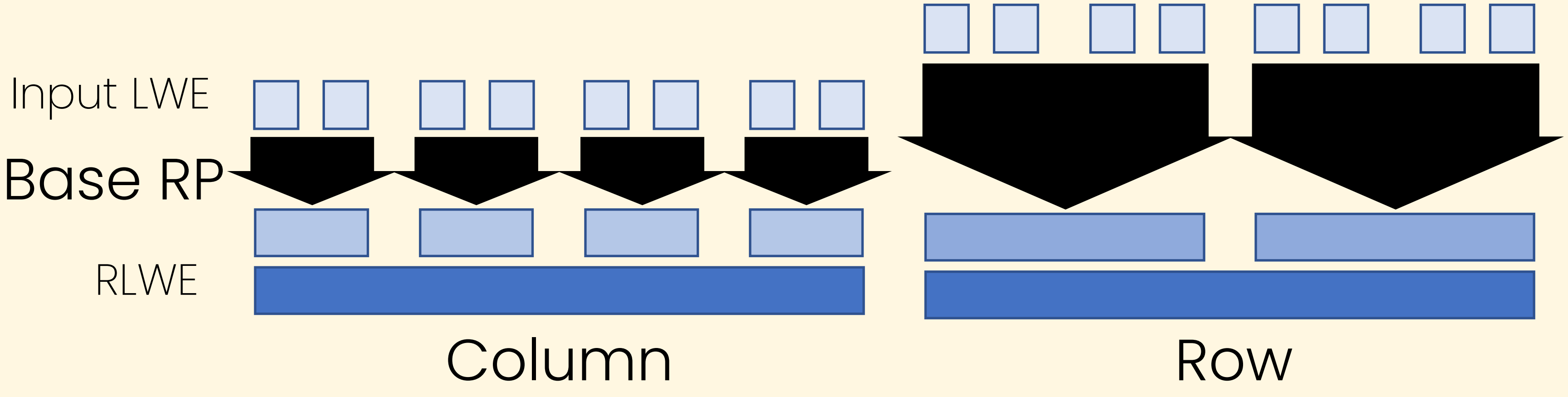
Base RP with Existing Approaches

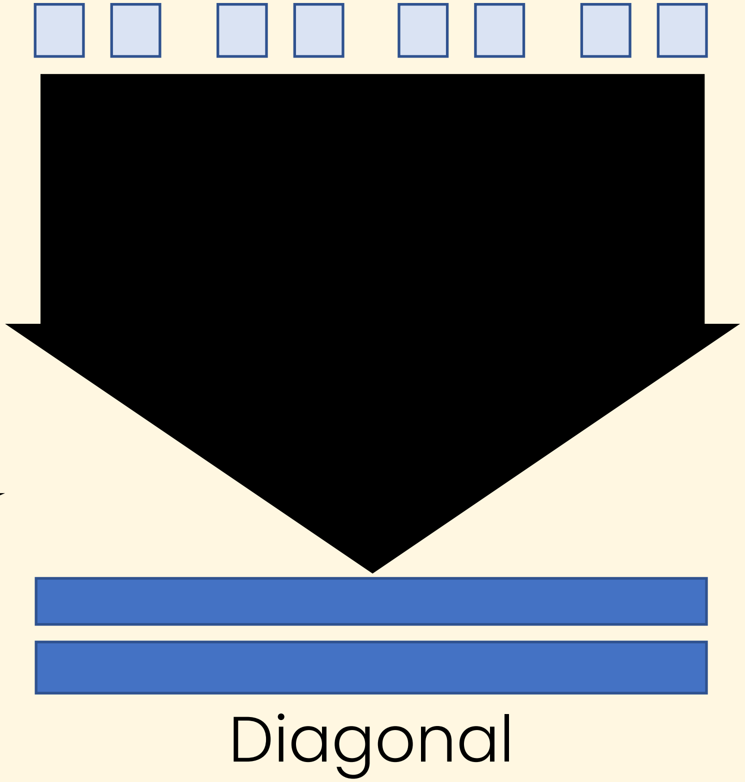


Base RP with Existing Approaches

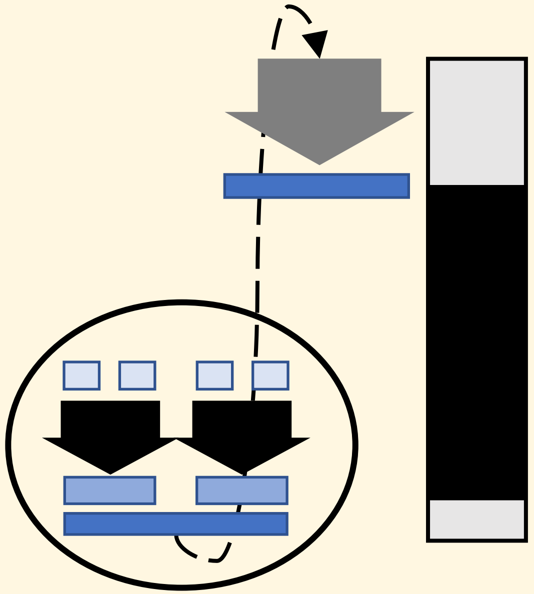


- The less moduli consumption is better.

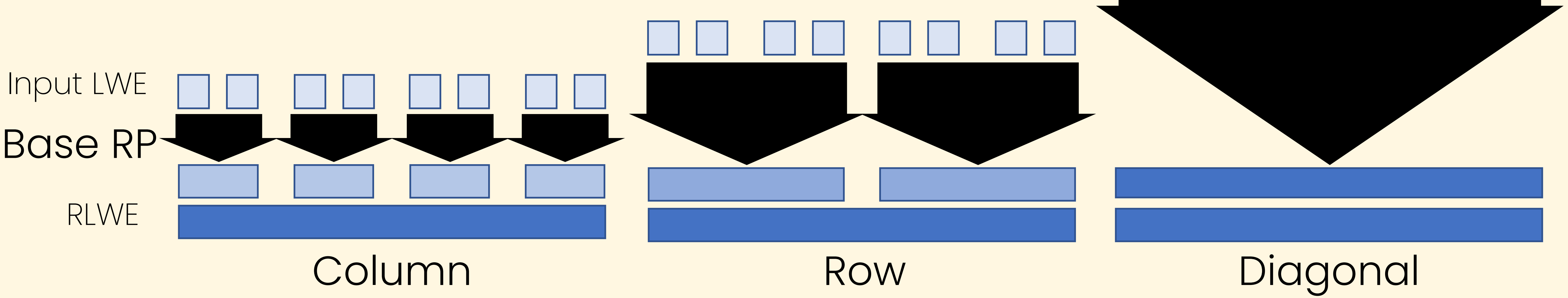




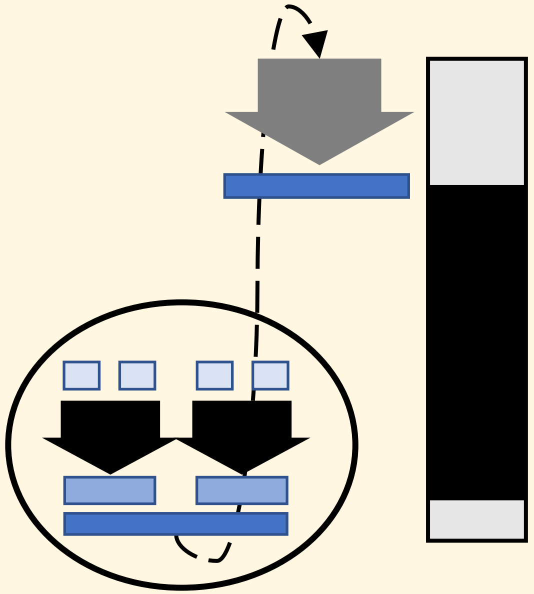
Base RP with Existing Approaches



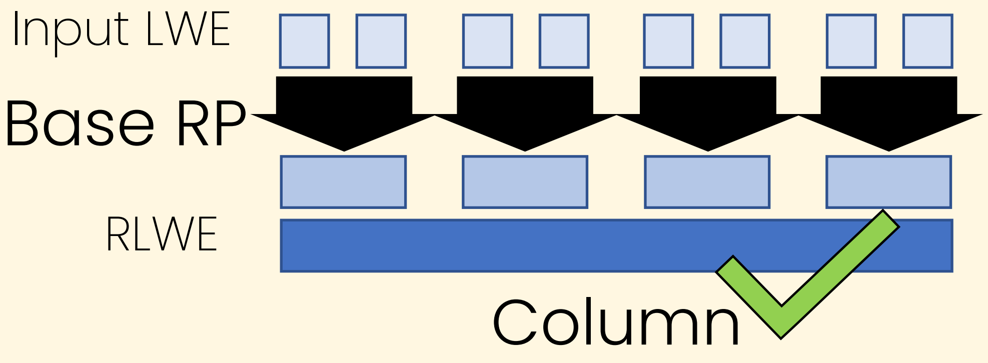
- The less moduli consumption is better.
- Column method is the most effective after optimizations.

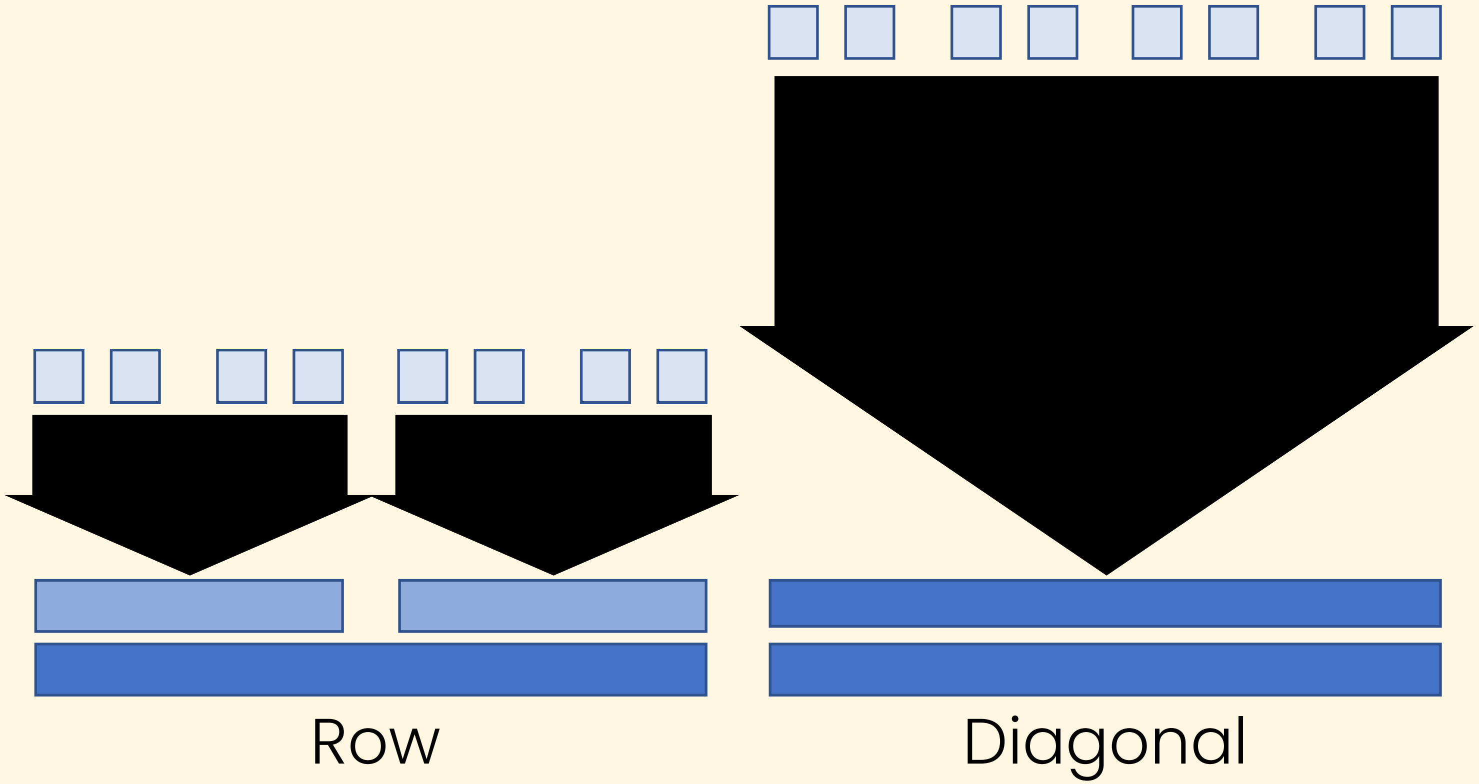


Base RP with Existing Approaches



- The less moduli consumption is better.
- Column method is the most effective after optimizations.

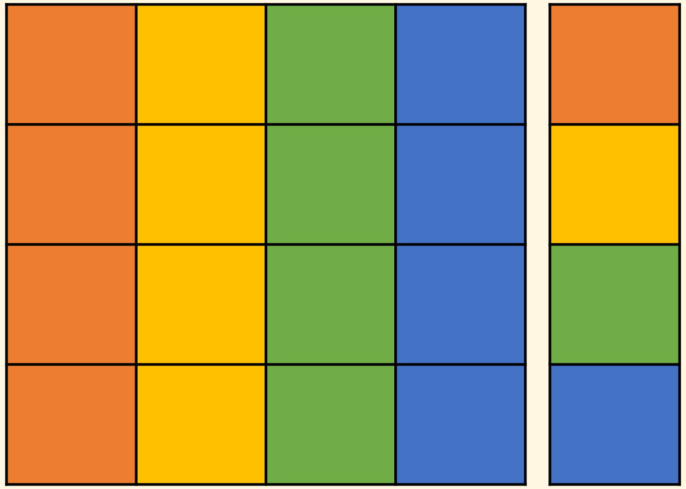




HERMES

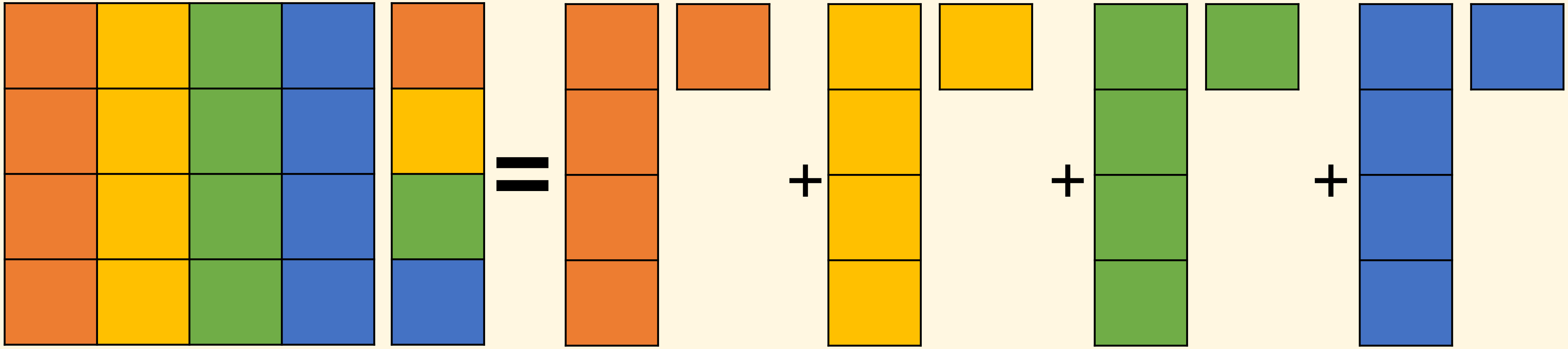
HERMES^0: Column method

HERMES^0: the column method with our optimizations



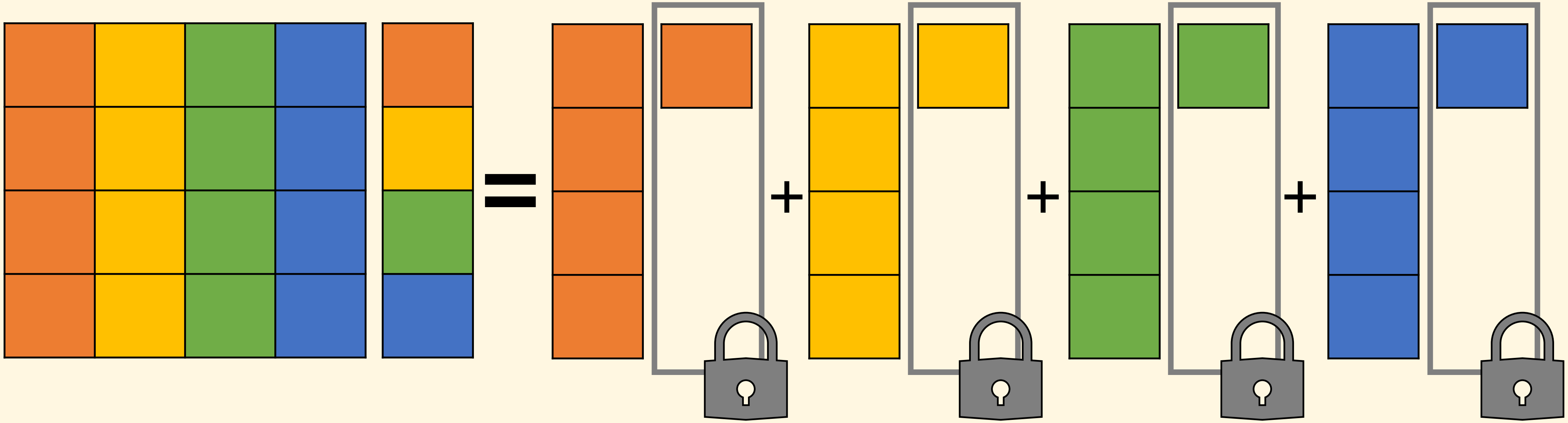
HERMES^0: Column method

HERMES^0: the column method with our optimizations



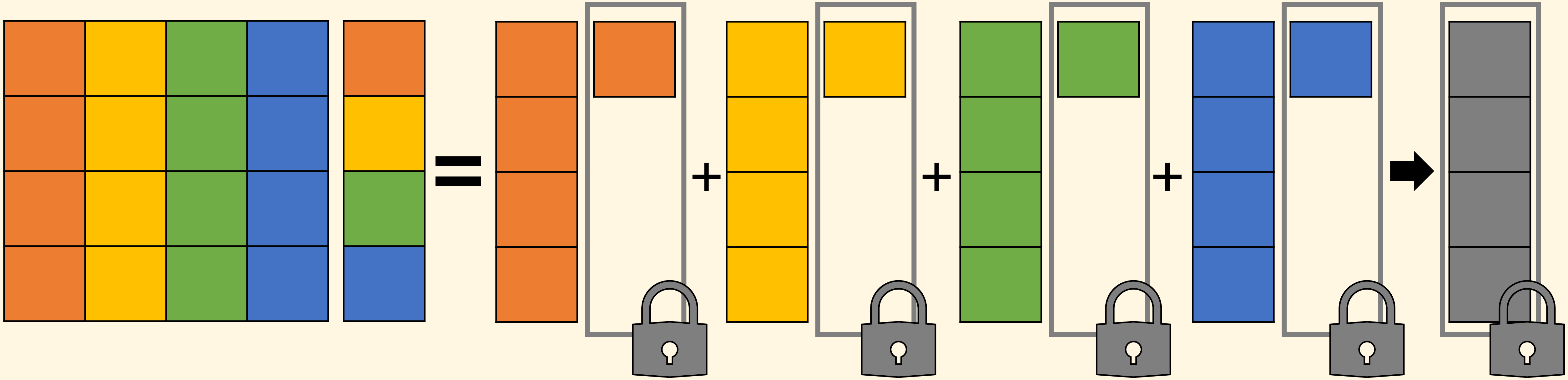
HERMES^0: Column method

HERMES^0: the column method with our optimizations



HERMES^0: Column method

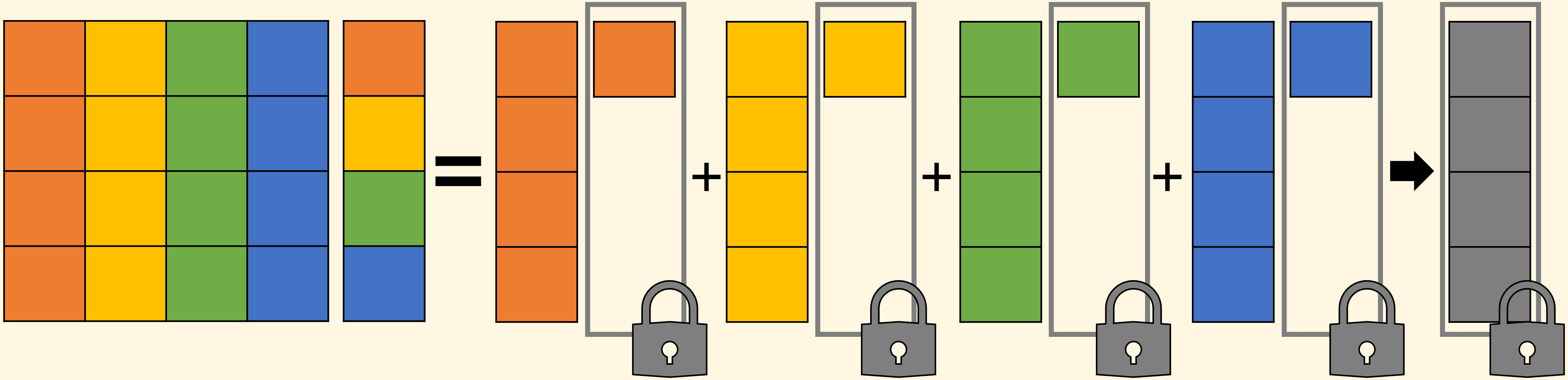
HERMES^0: the column method with our optimizations



HERMES^0: Column method

HERMES^0: the column method with our optimizations

- Practically fast.

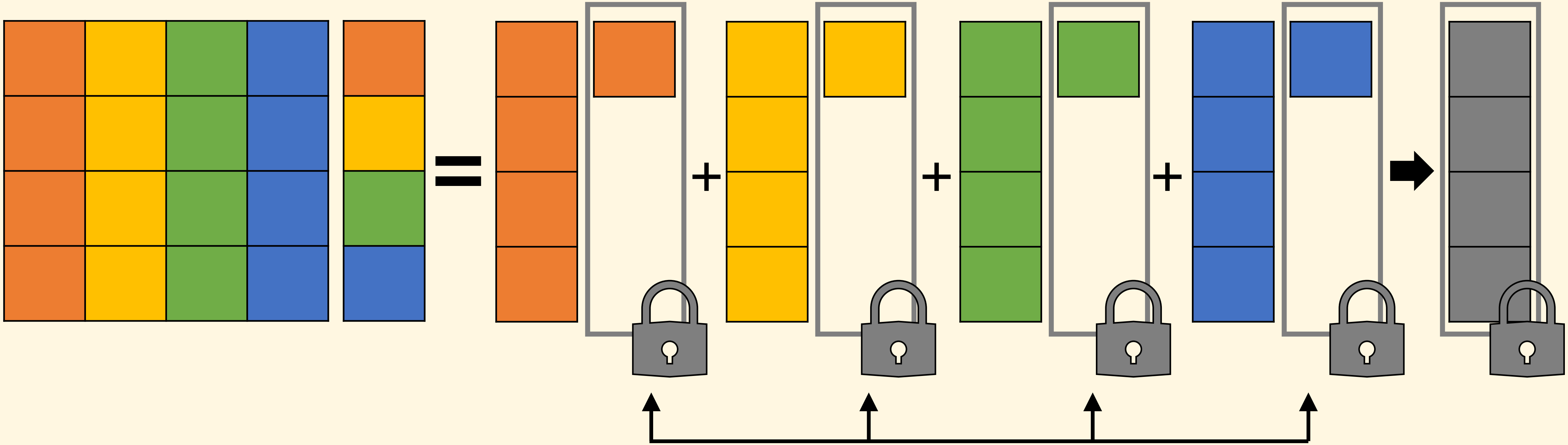


HERMES^0: Column method

HERMES^0: the column method with our optimizations

- Practically fast.

K: LWE dimension



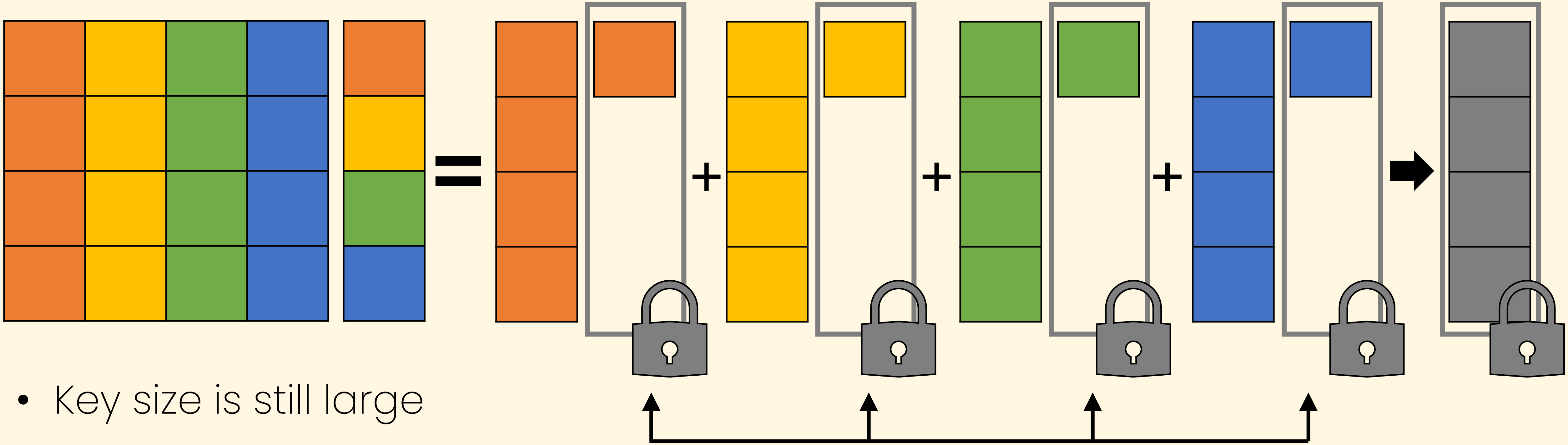
(Switching) Keys for BaseRP $= K$ switching keys

HERMES^0: Column method

HERMES^0: the column method with our optimizations

- Practically fast.

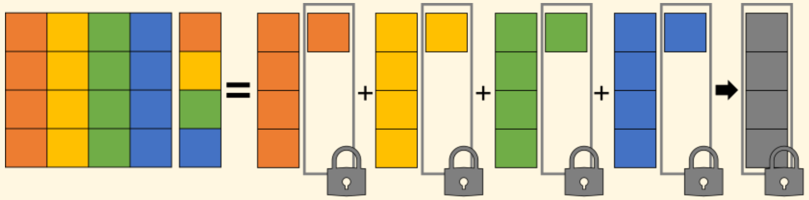
K: LWE dimension



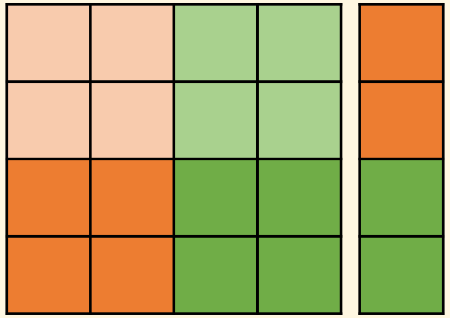
- Key size is still large

(Switching) Keys for BaseRP $= K$ switching keys

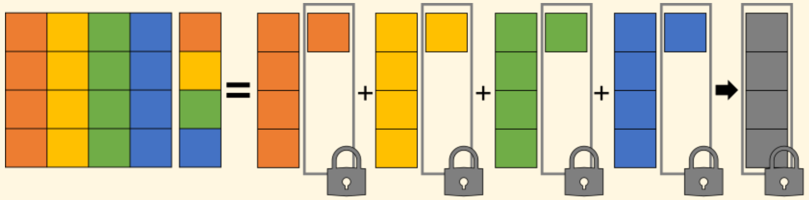
HERMES[1]: Block method



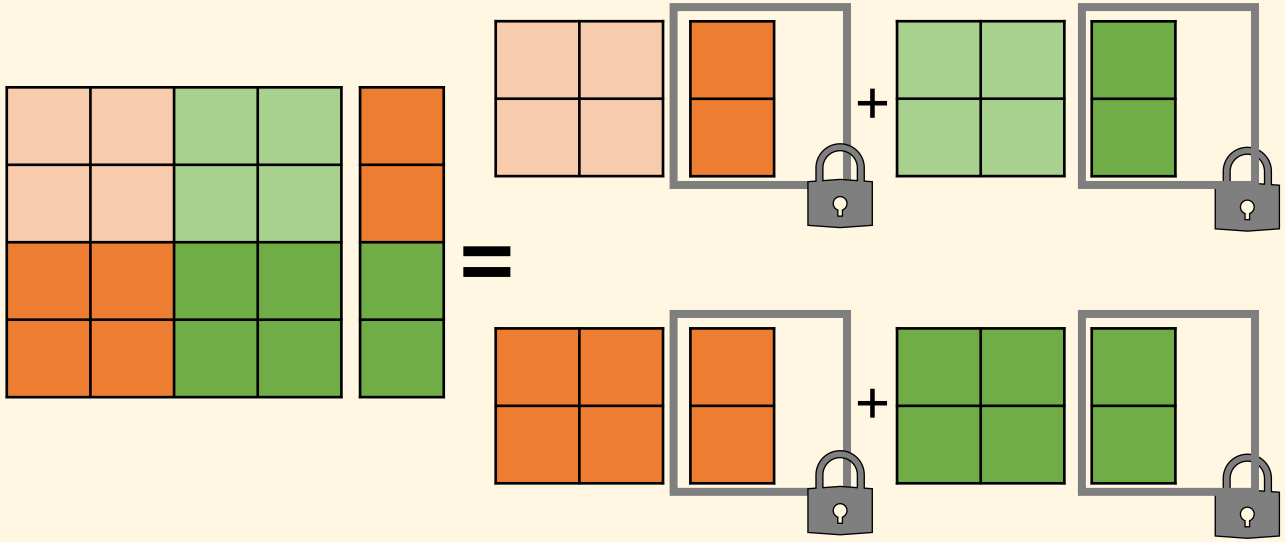
HERMES[1]: the *block method* with our optimizations



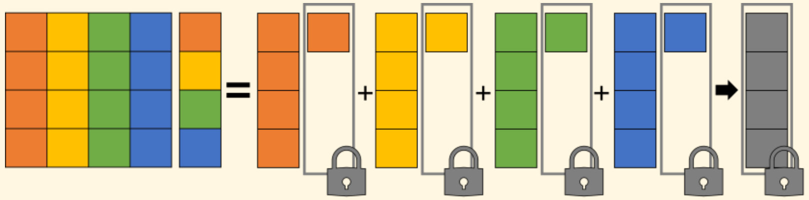
HERMES[1]: Block method



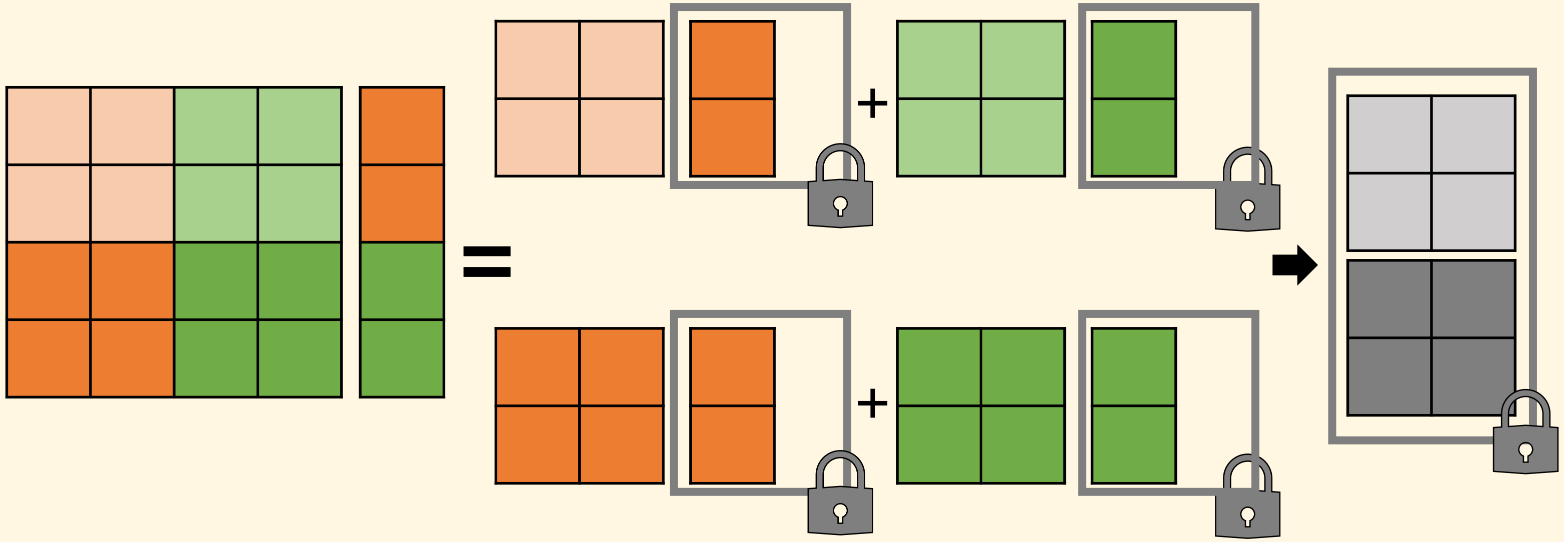
HERMES[1]: the *block method* with our optimizations



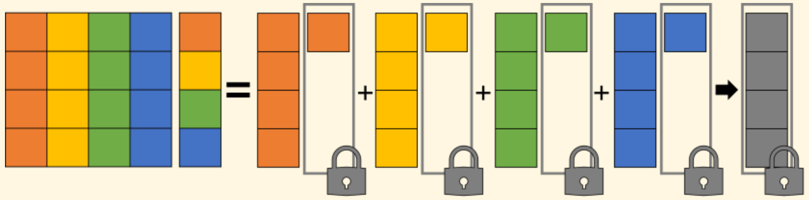
HERMES[1]: Block method



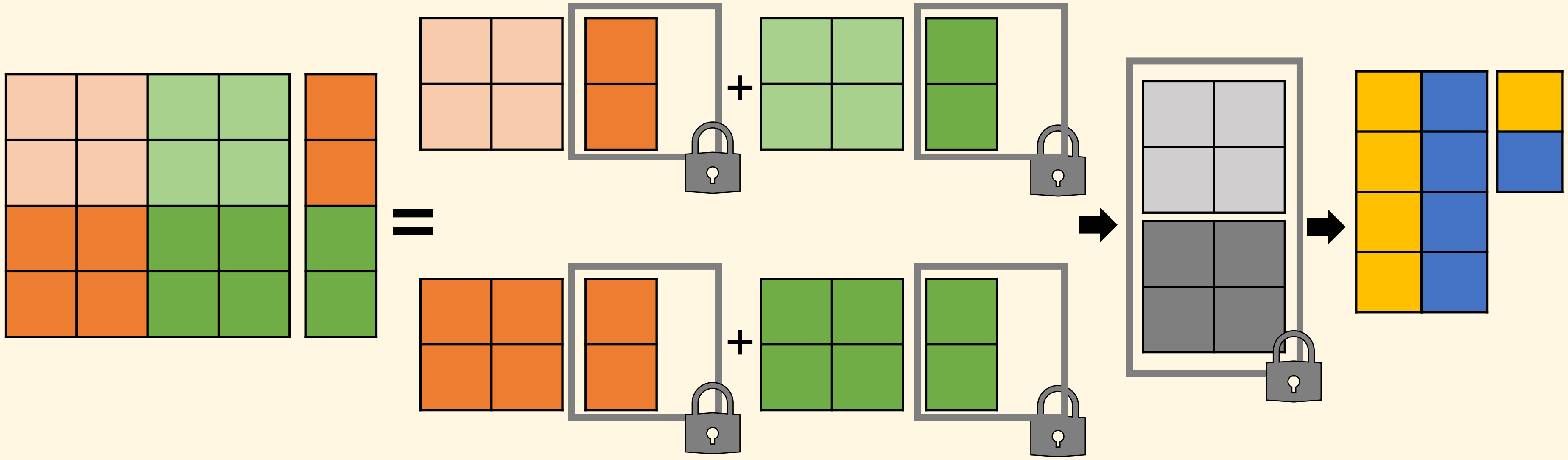
HERMES[1]: the *block method* with our optimizations



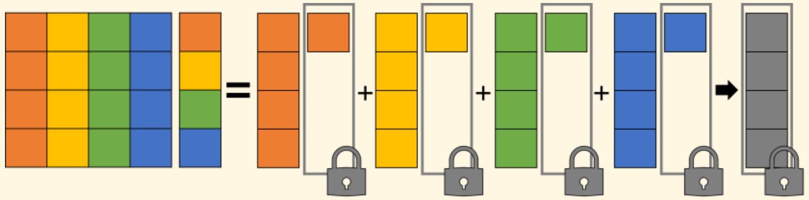
HERMES[1]: Block method



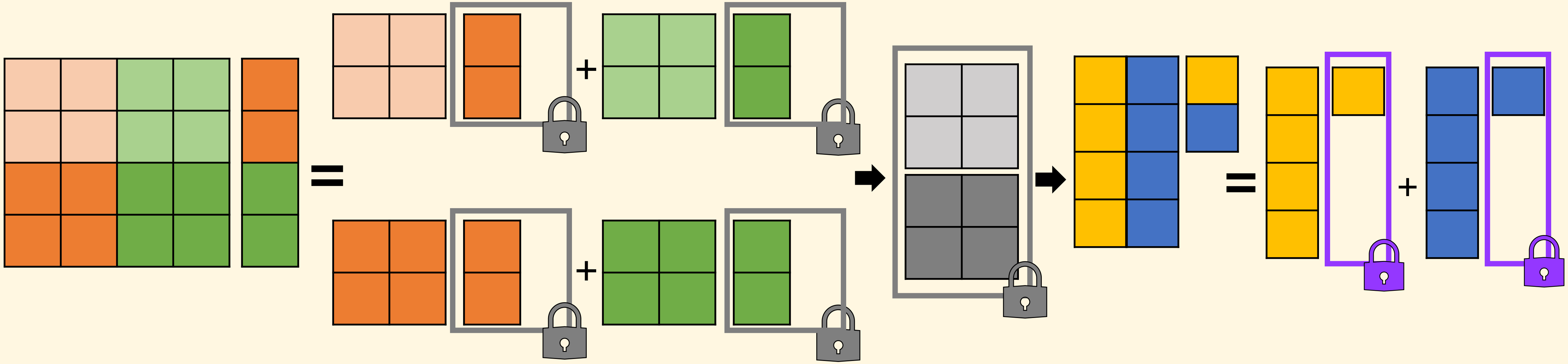
HERMES[1]: the *block method* with our optimizations



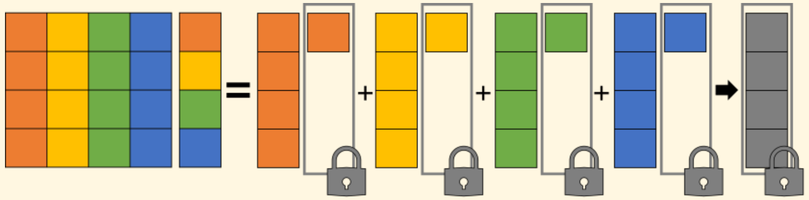
HERMES[1]: Block method



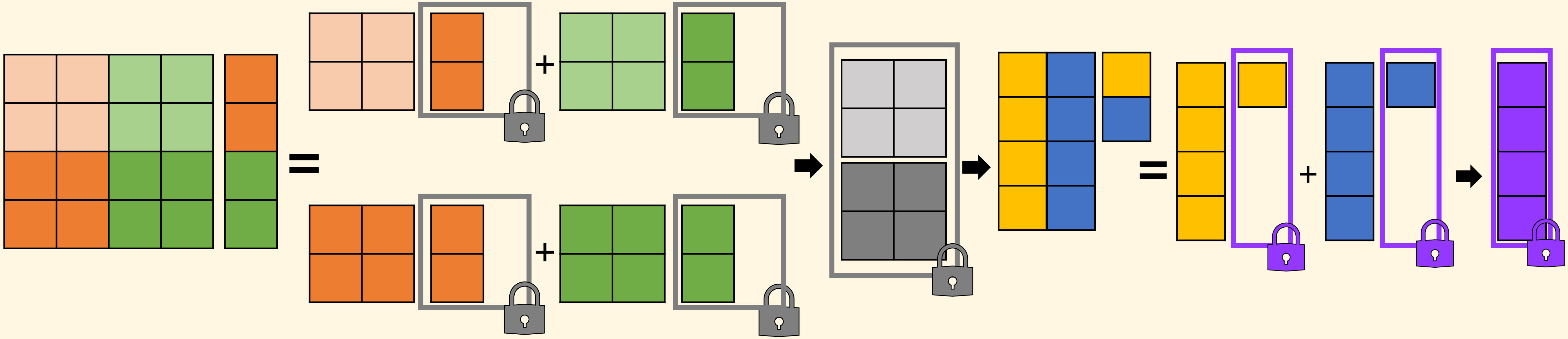
HERMES[1]: the *block method* with our optimizations



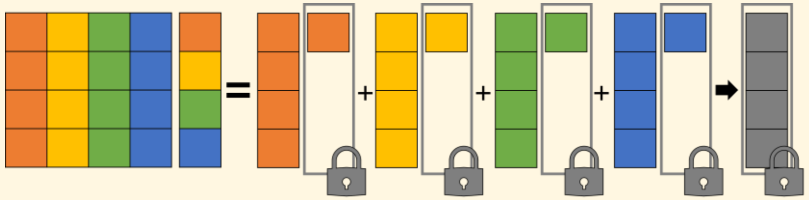
HERMES[1]: Block method



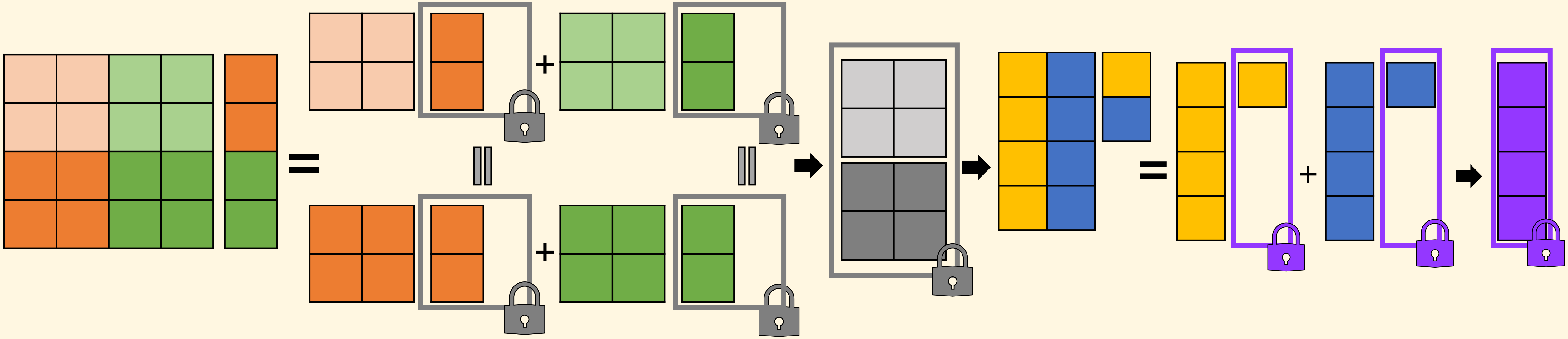
HERMES[1]: the *block method* with our optimizations



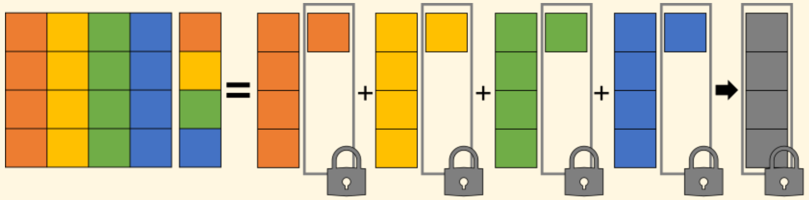
HERMES[1]: Block method



HERMES[1]: the *block method* with our optimizations

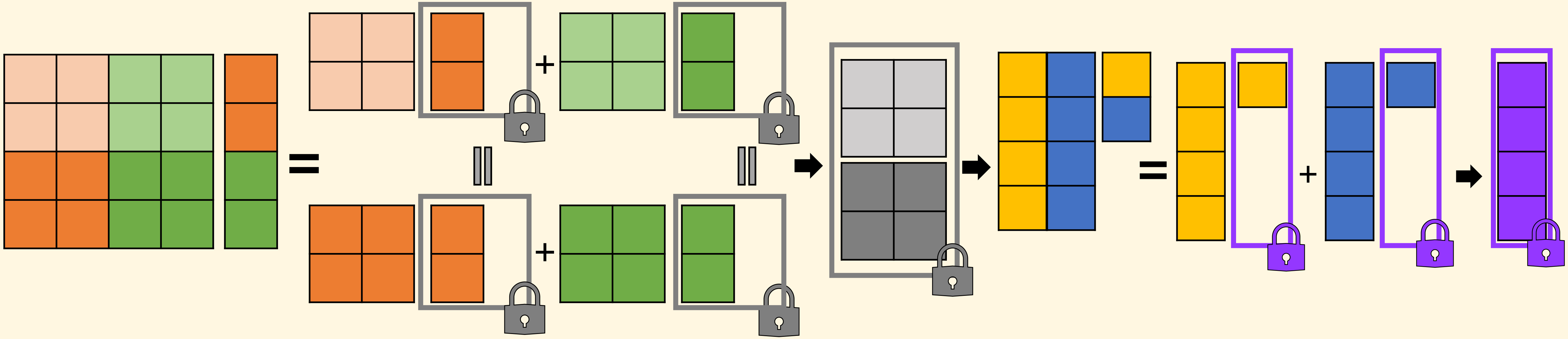


HERMES[1]: Block method



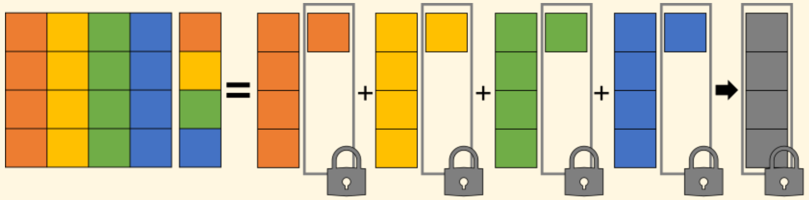
HERMES[1]: the *block method* with our optimizations

K: LWE dimension



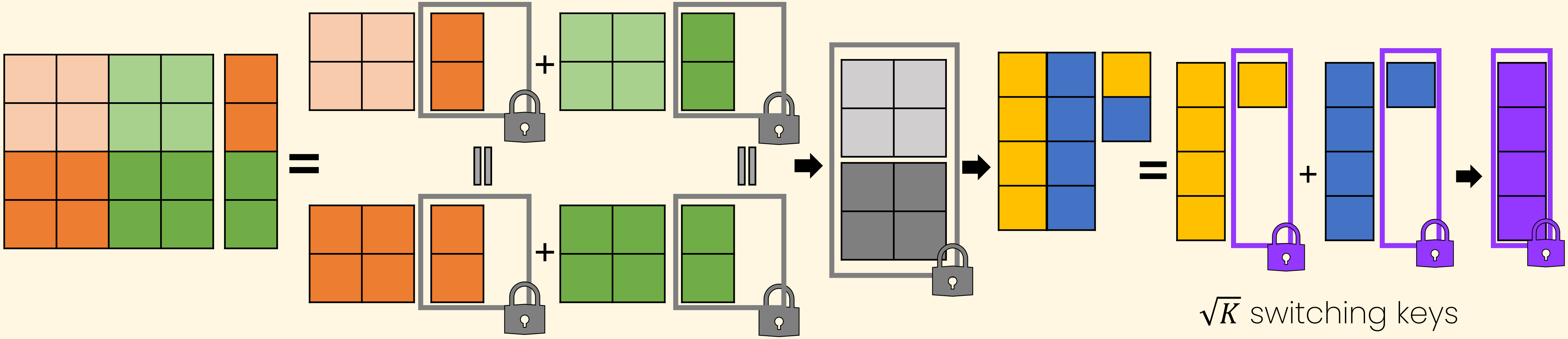
$\sqrt{K}$ switching keys

HERMES[1]: Block method



HERMES[1]: the *block method* with our optimizations

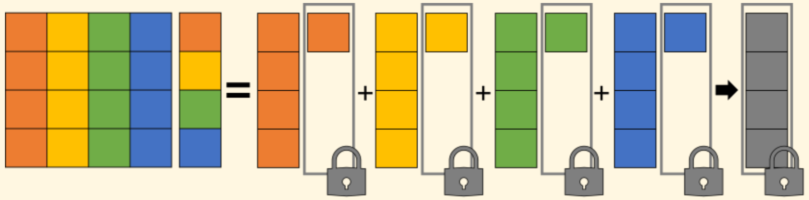
K: LWE dimension



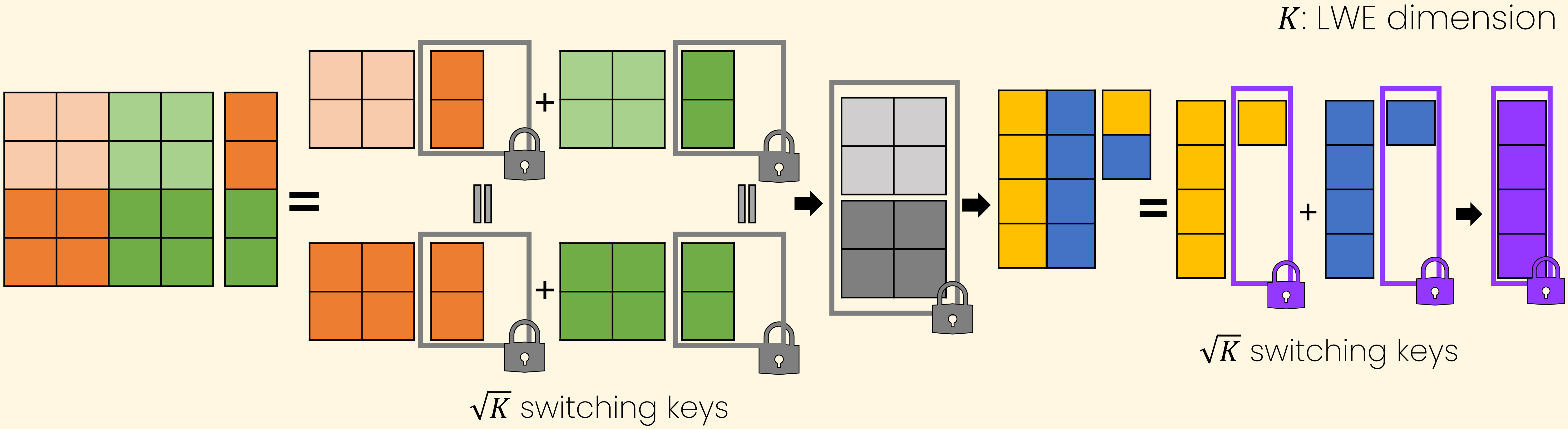
$\sqrt{K}$ switching keys

$\sqrt{K}$ switching keys

HERMES[1]: Block method

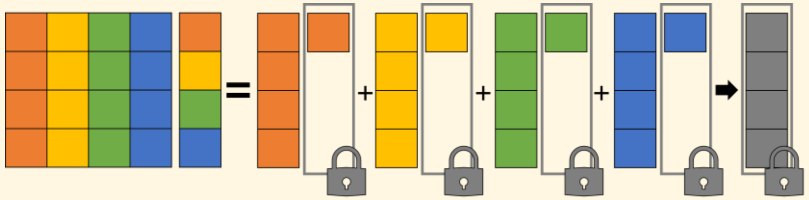


HERMES[1]: the *block method* with our optimizations

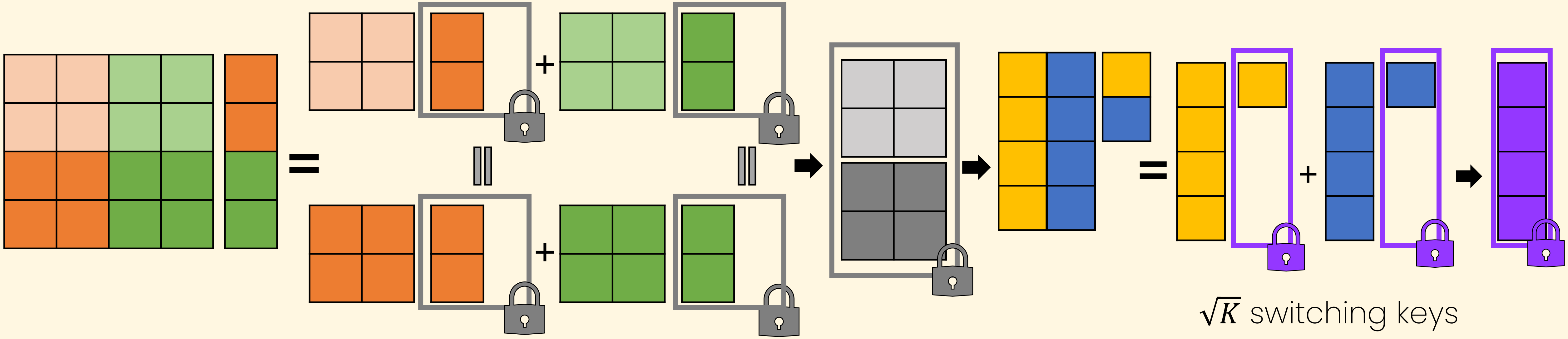


- How to encode the blocks?

HERMES[1]: Block method

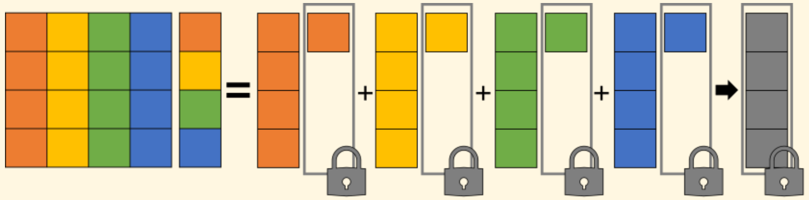


HERMES[1]: the *block method* with our optimizations



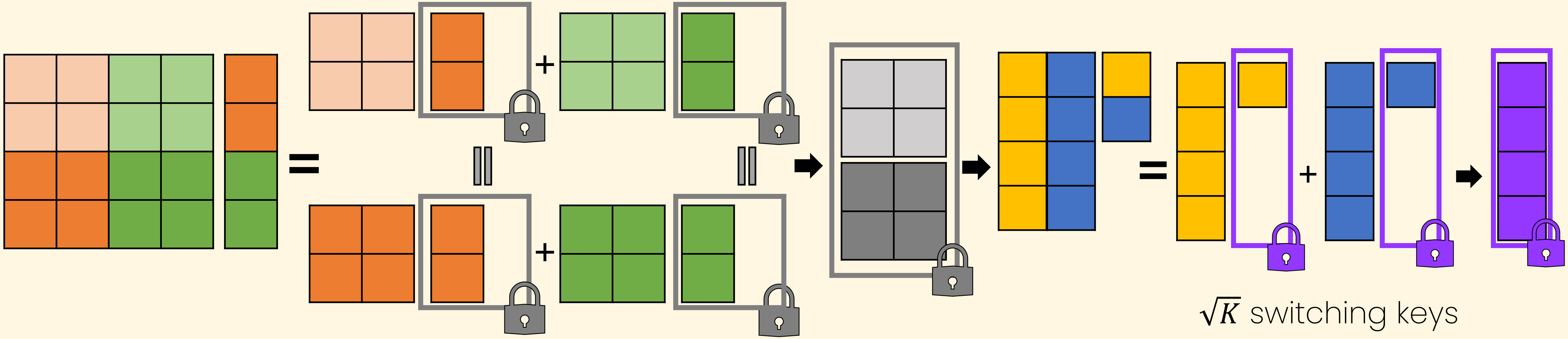
- How to encode the blocks?
- How to squeeze the blocks?

HERMES[1]: Block method



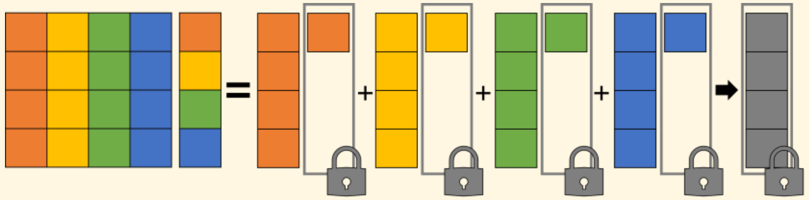
HERMES[1]: the *block method* with our optimizations

K: LWE dimension



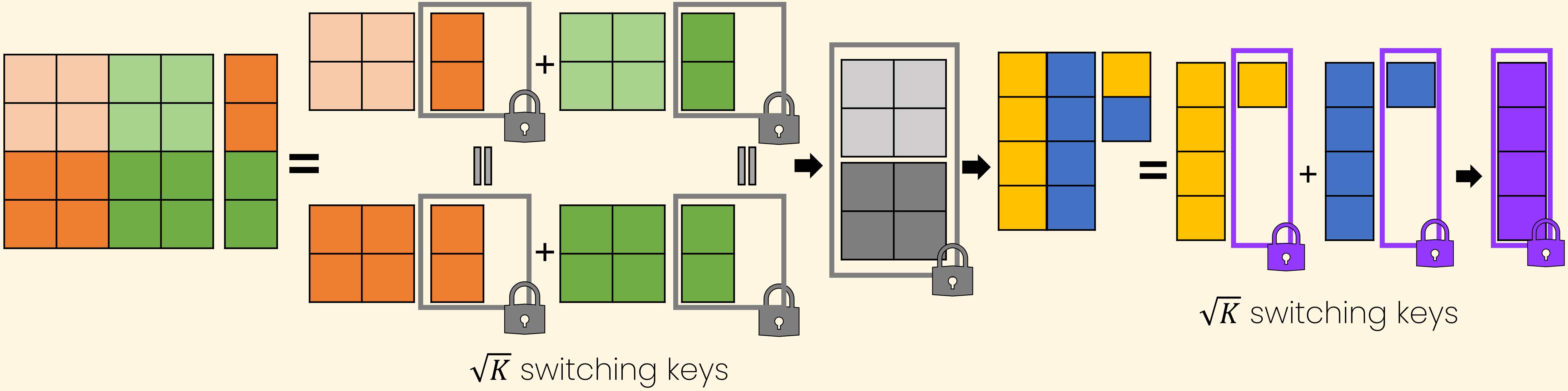
- How to encode the blocks? *Module LWE (MLWE)*
- How to squeeze the blocks?

HERMES[1]: Block method



HERMES[1]: the *block method* with our optimizations

K: LWE dimension



- How to encode the blocks? *Module LWE (MLWE)*
- How to squeeze the blocks? *MLWE key switching / MLWE ring switching*

Experimental Results

Ring Packing

<table>
<tr><th colspan="2" rowspan="2">FHE RP Method</th><th rowspan="2">RLWE modulus</th><th rowspan="2">RLWE degree</th><th rowspan="2"># of input LWE</th><th rowspan="2">Amortized Time (ms/slot)</th><th colspan="2">Key size (MB)</th></tr>
<tr><th>Base RP</th><th>Half-BTS</th></tr>
<tr><td>Pegasus [LHH+21]</td><td>Diagonal</td><td>2^{270}</td><td>2^{16}</td><td>2^{12}</td><td>12.62</td><td colspan="2">3540</td></tr>
<tr><td>[CDKS21] with Optimizations</td><td>Row</td><td>2^{562}</td><td>2^{16}</td><td>2^{16}</td><td>0.83</td><td>1</td><td>667</td></tr>
<tr><td>HERMES[0]</td><td>Column</td><td>2^{562}</td><td>2^{16}</td><td>2^{16}</td><td>0.44</td><td>114</td><td>667</td></tr>
<tr><td rowspan="2">HERMES[1]</td><td rowspan="2">Block</td><td>2^{562}</td><td>2^{16}</td><td>2^{16}</td><td>0.47</td><td>6</td><td>667</td></tr>
<tr><td>2^{270}</td><td>2^{15}</td><td>2^{15}</td><td>0.31</td><td>5</td><td>542</td></tr>
</table>

All experiments are measured on AMD® Ryzen 7 3700x 8-core processor with a single-threaded CPU.
Pegasus figures are borrowed from [LHH+21]; measured on single-threaded Intel Xeon Platinum 8269CY CPU (20-cores) at 2.50GHz.

Ring Packing

FHE RP Method		RLWE modulus	RLWE degree	# of input LWE	Amortized Time (ms/slot)	Key size (MB)	
						Base RP	Half-BTS
Pegasus [LHH+21]	Diagonal	2^{270}	2^{16}	2^{12}	12.62	3540	
[CDKS21] with Optimizations	Row	2^{562}	2^{16}	2^{16}	0.83	1	667
HERMES[0]	Column	2^{562}	2^{16}	2^{16}	0.44	114	667
HERMES[1]	Block	2^{562}	2^{16}	2^{16}	0.47	6	667
		2^{270}	2^{15}	2^{15}	0.31	5	542

All experiments are measured on AMD® Ryzen 7 3700x 8-core processor with a single-threaded CPU.
Pegasus figures are borrowed from [LHH+21]; measured on single-threaded Intel Xeon Platinum 8269CY CPU (20-cores) at 2.50GHz.

Ring Packing

FHE RP Method		RLWE modulus	RLWE degree	# of input LWE	Amortized Time (ms/slot)	Key size (MB)	
						Base RP	Half-BTS
Pegasus [LHH+21]	Diagonal	2^{270}	2^{16}	2^{12}	12.62	3540	
[CDKS21] with Optimizations	Row	2^{562}	2^{16}	2^{16}	0.83	1	667
HERMES[0]	Column	2^{562}	2^{16}	2^{16}	0.44	114	667
HERMES[1]	Block	2^{562}	2^{16}	2^{16}	0.47	6	667
		2^{270}	2^{15}	2^{15}	0.31	5	542

All experiments are measured on AMD® Ryzen 7 3700x 8-core processor with a single-threaded CPU.
Pegasus figures are borrowed from [LHH+21]; measured on single-threaded Intel Xeon Platinum 8269CY CPU (20-cores) at 2.50GHz.

Transciphering

Scheme	RLWE degree	Granularity	Latency (s)	Expansion Ratio	Mean Precision
HERA [CHK+21]	2^{16}	16	141.6	1.24	19.1
Rubato [HKL+22]		64	106.4	1.26	18.9
		16	71.1	1.31	18.8
HERMES		64	25.8	1.58	23.0
		16	25.7	1.58	23.0
		1	30.9	1.58	23.0

All experiments are measured on AMD® Ryzen 7 3700x 8-core processor with a single-threaded CPU.
HERA and Rubato figures are borrowed from [CHK+21] and [HKL+22]; measured on AMD Ryzen 7 2700X @ 3.70 GHz single-threaded CPU.

Wrapping up!

Wrapping up!

- Optimize the existing RP methods for FHE RP.

Wrapping up!

- Optimize the existing RP methods for FHE RP.
- Propose a new efficient FHE RP method, HERMES.
  - Compared to state-of-the-art, 40x faster in terms of throughput.

Wrapping up!

- Optimize the existing RP methods for FHE RP.
- Propose a new efficient FHE RP method, HERMES.
  - Compared to state-of-the-art, 40x faster in terms of throughput.
- Application to transciphering.

Wrapping up!

- Optimize the existing RP methods for FHE RP.
- Propose a new efficient FHE RP method, HERMES.
    - Compared to state-of-the-art, 40x faster in terms of throughput.
- Application to transciphering.

eprint: 2023/1244

Thank you!

References

[BGGJ20] C. Boura, N. Gama, M. Georgieva, and D. Jetchev. CHIMERA: combining ring-LWE-based fully homomorphic encryption schemes. J. Math. Cryptol., 2020.

[CDKS21] H. Chen, W. Dai, M. Kim, and Y. Song. Efficient homomorphic conversion between (ring) LWE ciphertexts. In ACNS, 2021.

[CGGI17] I. Chillotti, N. Gama, M. Georgieva, and M. Izabach`ene. Faster packed homomorphic operations and efficient circuit bootstrapping for TFHE. In ASIACRYPT, 2017

[CHK+21] J. Cho, J. Ha, S. Kim, B. Lee, J. Lee, J. Lee, D. Moon, and H. Yoon. Transciphering framework for approximate homomorphic encryption. In ASIACRYPT, 2021.

[GHPS13] C. Gentry, S. Halevi, C. Peikert, and N. P. Smart. Field switching in BGV-style homomorphic encryption. Journal of Computer Security, 2013.

[HKL+22] J. Ha, S. Kim, B. Lee, J. Lee, and M. Son. Rubato: Noisy ciphers for approximate homomorphic encryption. In EUROCRYPT, 2022.

[HS14] S. Halevi and V. Shoup. Algorithms in HElib. In CRYPTO, 2014.

[LHH+21] W.-J. Lu, Z. Huang, C. Hong, Y. Ma, and H. Qu. PEGASUS: bridging polynomial and non-polynomial evaluations in homomorphic encryption. In S&P, 2021.

[MS18] D. Micciancio and J. Sorrell. Ring packing and amortized FHEW bootstrapping. In ICALP, 2018.