Digital Signatures for Authenticating Compressed JPEG Images

Simon Erfurth

University of Southern Denmark, Denmark

simon@serfurth.dk www.serfurth.dk @SimonErfurth

Motivating Example: Mitigating Fake News

Motivating Example: Mitigating Fake News

Problem

- More and more news content is consumed on social media.
  $\Rightarrow$ Users don't know/remember the source.

Motivating Example: Mitigating Fake News

Problem

- More and more news content is consumed on social media.
  $\Rightarrow$ Users don't know/remember the source.
- News media's reputation is an important heuristic.
  $\Rightarrow$ Harder to tell apart real and fake news.

Motivating Example: Mitigating Fake News

Problem

- More and more news content is consumed on social media.
  $\Rightarrow$ Users don't know/remember the source.
- News media's reputation is an important heuristic.
  $\Rightarrow$ Harder to tell apart real and fake news.
- GenAI lowers the bar for creating visual misinformation.
  $\Rightarrow$ Misinformation might drown out authentic content.

Motivating Example: Mitigating Fake News

Problem

- More and more news content is consumed on social media.
  $\Rightarrow$ Users don't know/remember the source.
- News media's reputation is an important heuristic.
  $\Rightarrow$ Harder to tell apart real and fake news.
- GenAI lowers the bar for creating visual misinformation.
  $\Rightarrow$ Misinformation might drown out authentic content.

We can use cryptography to fight this!

Motivating Example: Mitigating Fake News

A (part of the) solution

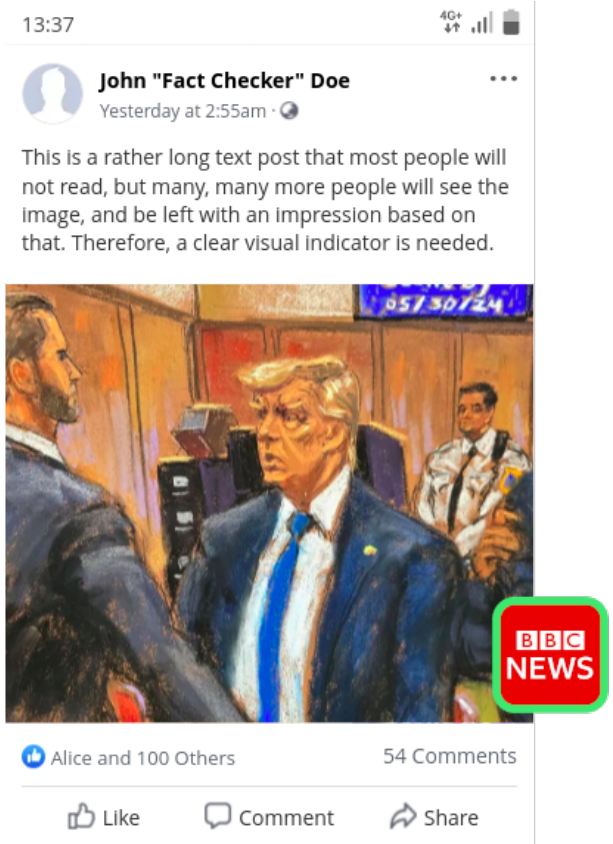
Add digital signatures and authenticity indicators.

Motivating Example: Mitigating Fake News

A (part of the) solution

Add digital signatures and authenticity indicators.

@EliotHiggins



[EH24]

But. . .

[PLZ+09]

But. . .

But. . .

But. . .

Problem

Standard digital signatures do not work with (lossy) compression.

Digital Signatures for Authenticating **Compressed** JPEG Images

Simon Erfurth

University of Southern Denmark, Denmark

simon@serfurth.dk www.serfurth.dk @SimonErfurth

Goal

Digital Signature Scheme Allowing (some) Compression

- $\mathsf{sk}, \mathsf{pk} \leftarrow \mathsf{KeyGen}(1^\lambda)$
- $s \leftarrow \mathsf{Sign}_{\mathsf{sk}}(i)$
- $i', s' \leftarrow \mathsf{Compress}(i, s, P)$
  Does not require sk
- $0/1 \leftarrow \mathsf{Verify}_{\mathsf{pk}}(i, s)$

Goal

Digital Signature Scheme Allowing (some) Compression

- $\mathsf{sk}, \mathsf{pk} \leftarrow \mathsf{KeyGen}(1^\lambda)$
- $s \leftarrow \mathsf{Sign}_{\mathsf{sk}}(i)$
- $i', s' \leftarrow \mathsf{Compress}(i, s, P)$
  Does not require sk
- $0/1 \leftarrow \mathsf{Verify}_{\mathsf{pk}}(i, s)$

Trivial Construction

Just provide the missing data.

Goal

Digital Signature Scheme Allowing (some) Compression

- $\mathsf{sk}, \mathsf{pk} \leftarrow \mathsf{KeyGen}(1^\lambda)$
- $s \leftarrow \mathsf{Sign}_{\mathsf{sk}}(i)$
- $i', s' \leftarrow \mathsf{Compress}(i, s, P)$
  Does not require sk
- $0/1 \leftarrow \mathsf{Verify}_{\mathsf{pk}}(i, s)$

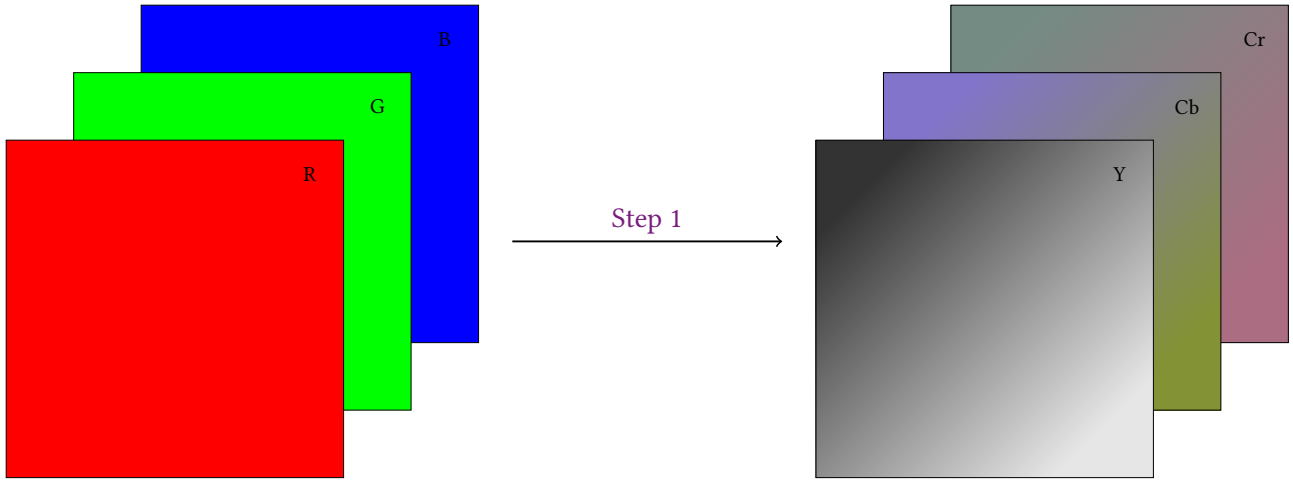
Trivial Construction

Just provide the missing data.
$\Rightarrow$ (lossy) compression is pointless.

JPEG Compression

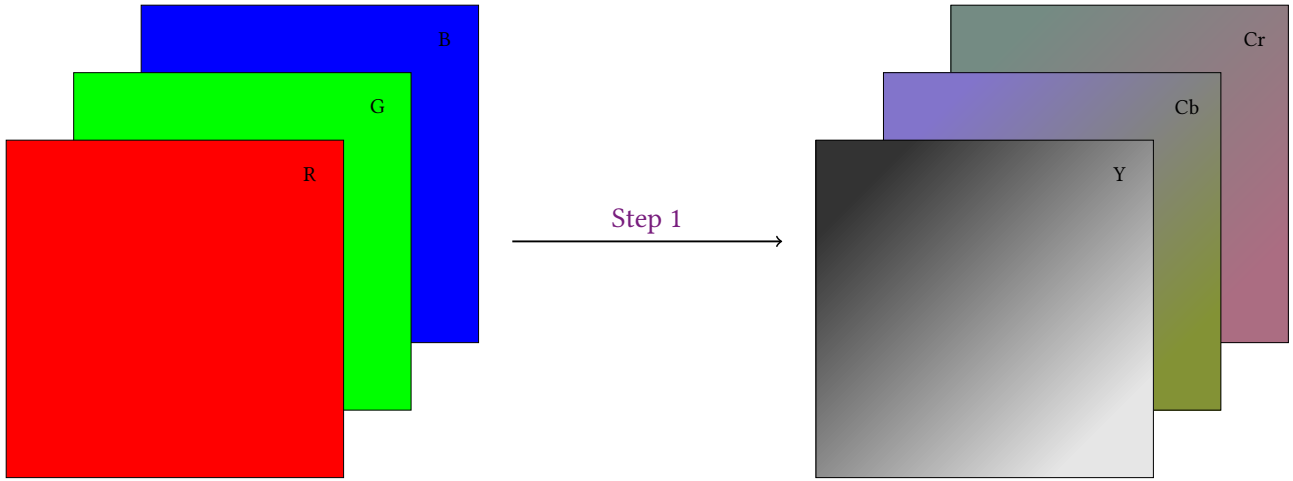
JPEG Compression

1. RGB to YCbCr color space



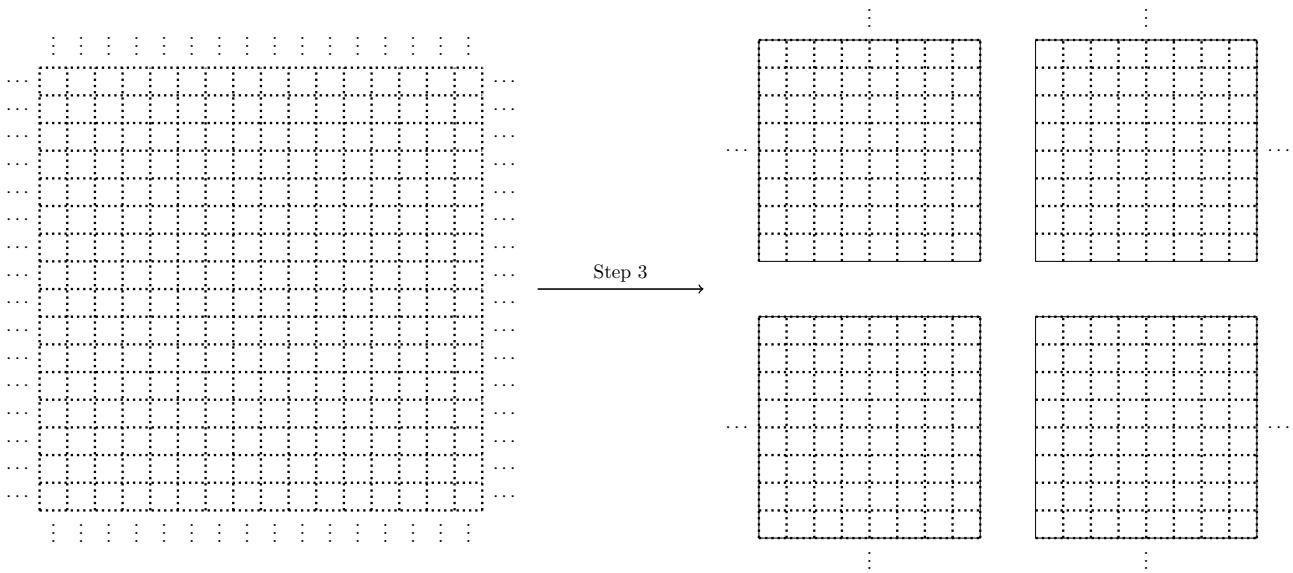
JPEG Compression

1. RGB to YCbCr color space
2. Optional: Down sample



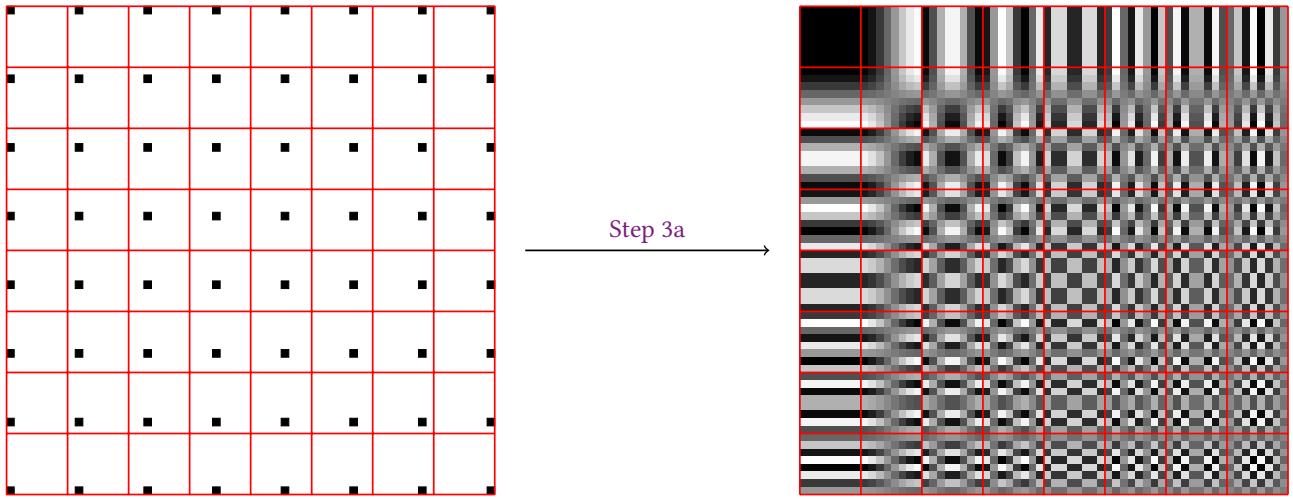
JPEG Compression

1. RGB to YCbCr color space
2. Optional: Down sample
3. Split into 8×8



JPEG Compression

1. RGB to YCbCr color space
2. Optional: Down sample
3. Split into 8×8
   a. Discrete cosine transformation



JPEG Compression

1. RGB to YCbCr color space
2. Optional: Down sample
3. Split into 8×8
   a. Discrete cosine transformation
   b. Quantization

815	-15	-10	-2	-2	-3	3	3
-3	2	-9	-4	4	-2	-2	3
0	1	-2	-4	-2	-2	2	-1
-3	-1	-3	3	1	0	3	1
0	-1	2	3	-1	1	-1	1
-5	6	1	-1	4	-1	-1	-3
2	1	2	0	0	7	3	-1
0	1	3	-3	-2	3	1	-5

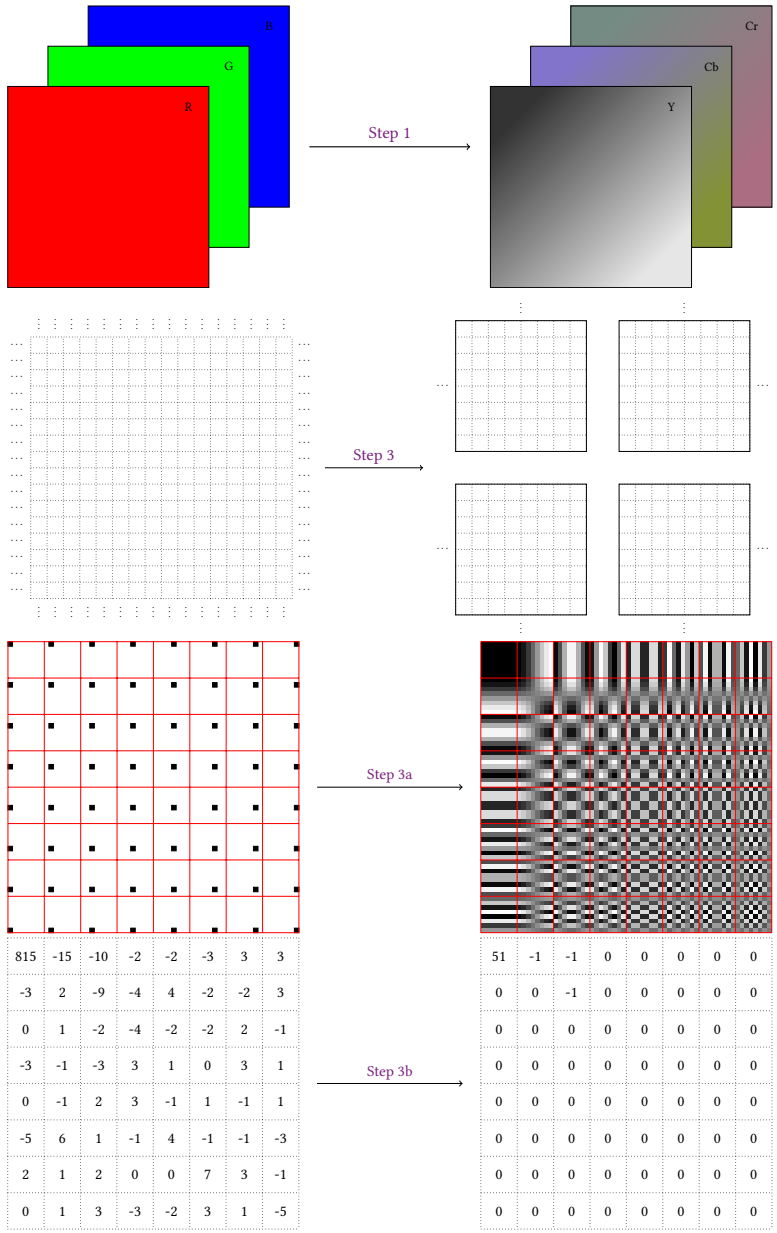
Step 3b →

51	-1	-1	0	0	0	0	0
0	0	-1	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

$$Q_{50} = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

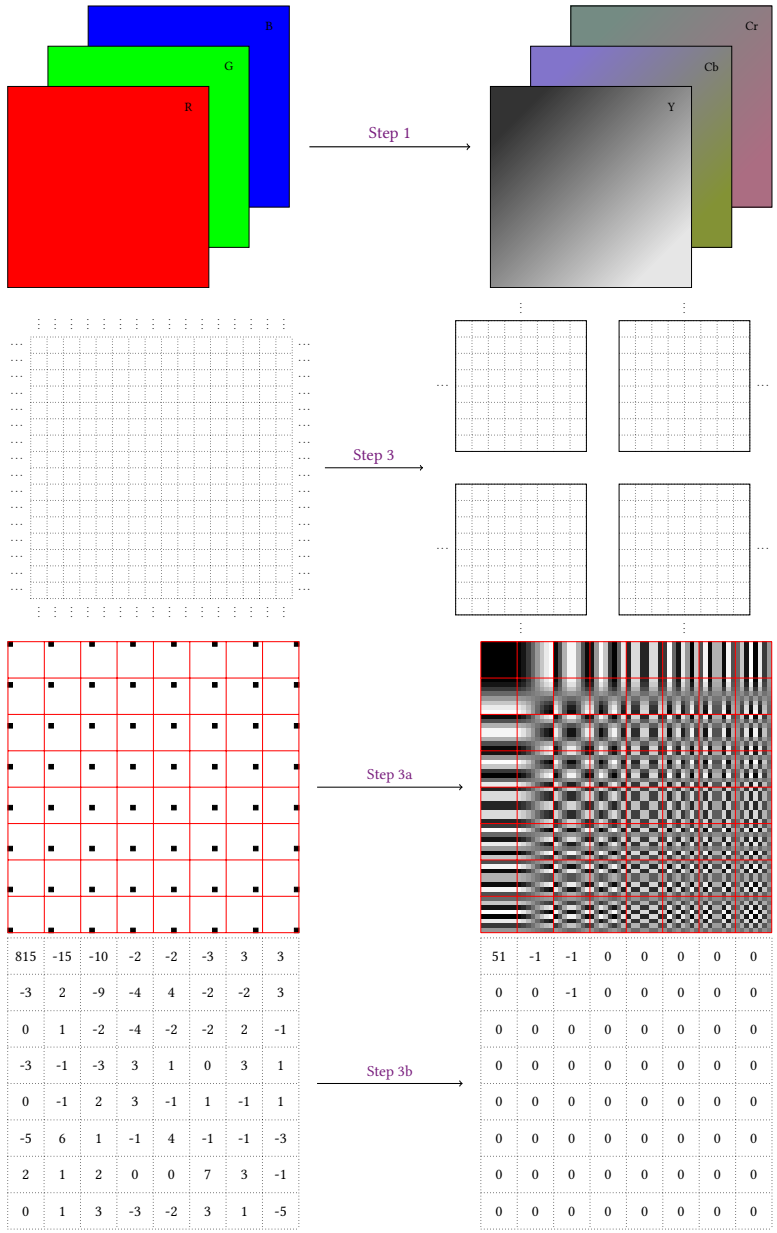
JPEG Compression

1. RGB to YCbCr color space
2. Optional: Down sample
3. Split into 8×8
   a. Discrete cosine transformation
   b. Quantization
4. Encode using entropy encoder



JPEG Compression

1. RGB to YCbCr color space
2. Optional: Down sample
3. Split into 8×8
   a. Discrete cosine transformation
   b. Quantization
4. Encode using entropy encoder



Construction: 1 Byte

Construction: 1 Byte

Observation: Quantization with 2^n is truncation

$$\frac{b_7\, b_6\, b_5\, b_4\, b_3\, b_2\, b_1\, b_0}{2^3} \rightsquigarrow b_7\, b_6\, b_5\, b_4\, b_3\, x x x$$

Construction: 1 Byte

Observation: Quantization with 2^n is truncation

$$\frac{b_7\, b_6\, b_5\, b_4\, b_3\, b_2\, b_1\, b_0}{2^3} \rightsquigarrow b_7\, b_6\, b_5\, b_4\, b_3\, x x x$$

Idea: Provide something instead of xxx?

Construction: 1 Byte

Observation: Quantization with 2^n is truncation

$$\frac{b_7\, b_6\, b_5\, b_4\, b_3\, b_2\, b_1\, b_0}{2^3} \rightsquigarrow b_7\, b_6\, b_5\, b_4\, b_3\, x x x$$

Idea: Provide something instead of xxx?

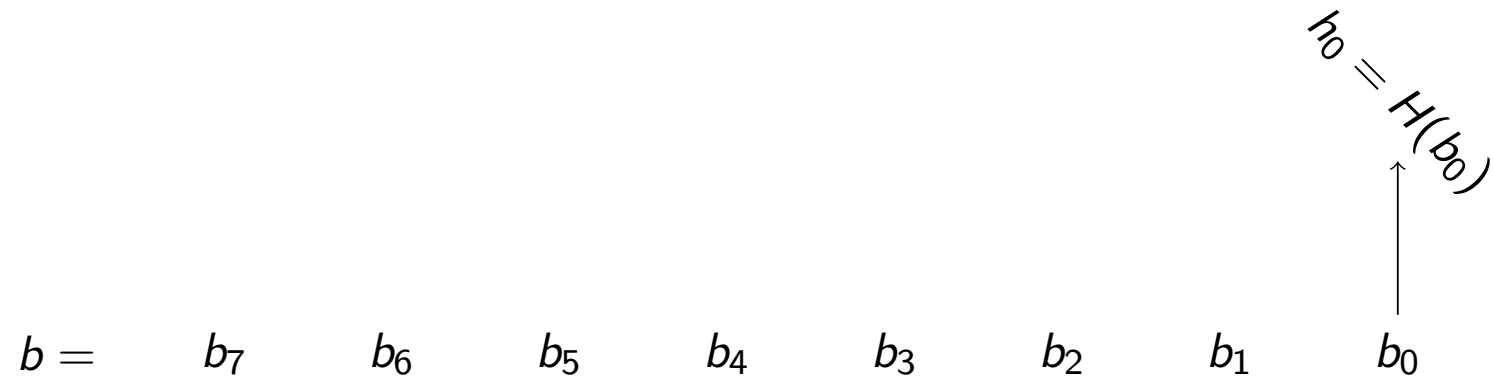
$$b = \quad b_7 \quad b_6 \quad b_5 \quad b_4 \quad b_3 \quad b_2 \quad b_1 \quad b_0$$

Construction: 1 Byte

Observation: Quantization with 2^n is truncation

$$\frac{b_7\, b_6\, b_5\, b_4\, b_3\, b_2\, b_1\, b_0}{2^3} \rightsquigarrow b_7\, b_6\, b_5\, b_4\, b_3\, x x x$$

Idea: Provide something instead of xxx?

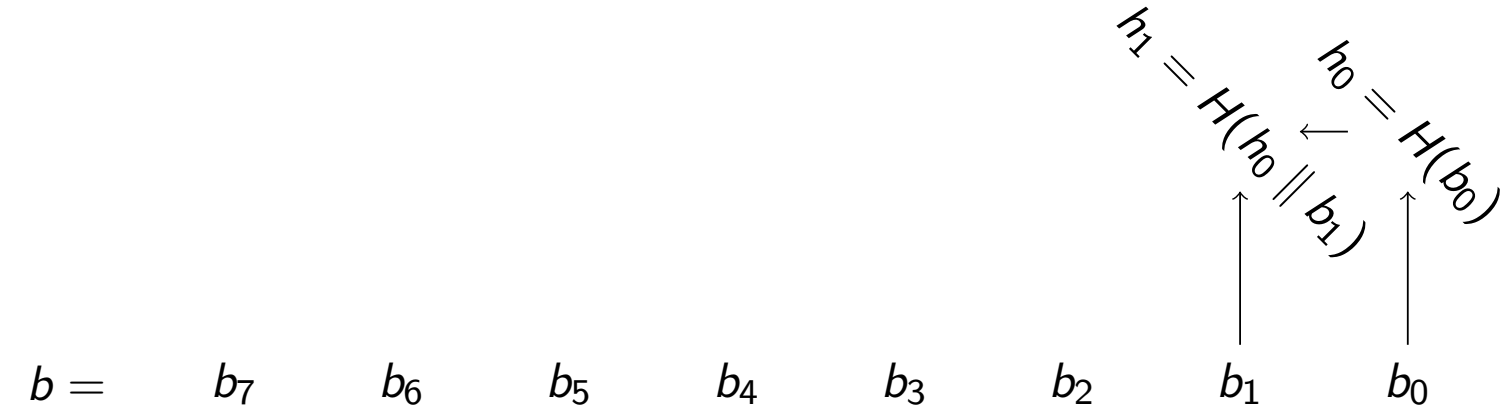


Construction: 1 Byte

Observation: Quantization with 2^n is truncation

$$\frac{b_7\, b_6\, b_5\, b_4\, b_3\, b_2\, b_1\, b_0}{2^3} \rightsquigarrow b_7\, b_6\, b_5\, b_4\, b_3\, x x x$$

Idea: Provide something instead of xxx?

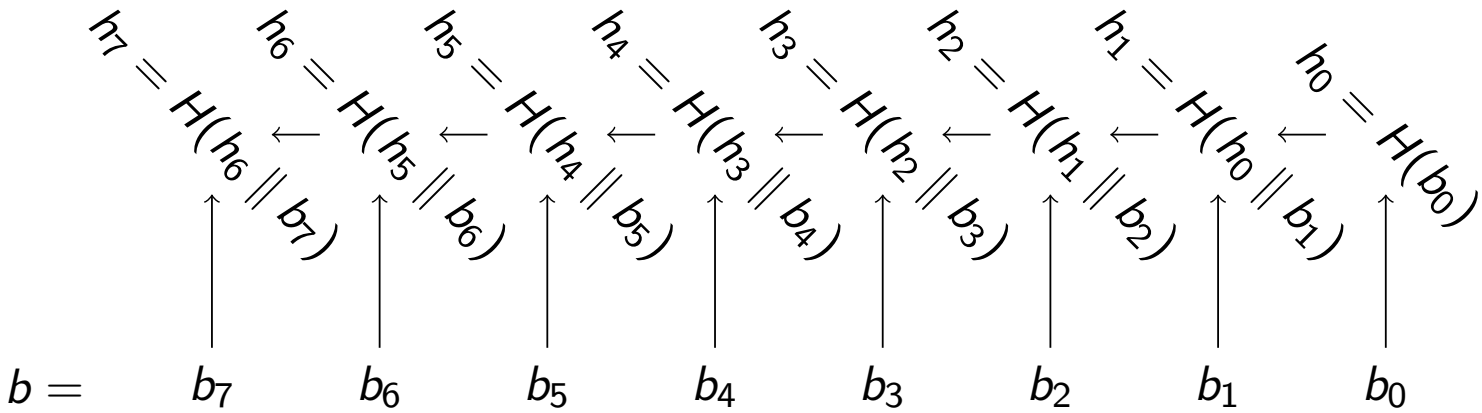


Construction: 1 Byte

Observation: Quantization with 2^n is truncation

$$\frac{b_7\, b_6\, b_5\, b_4\, b_3\, b_2\, b_1\, b_0}{2^3} \rightsquigarrow b_7\, b_6\, b_5\, b_4\, b_3\, x x x$$

Idea: Provide something instead of xxx?

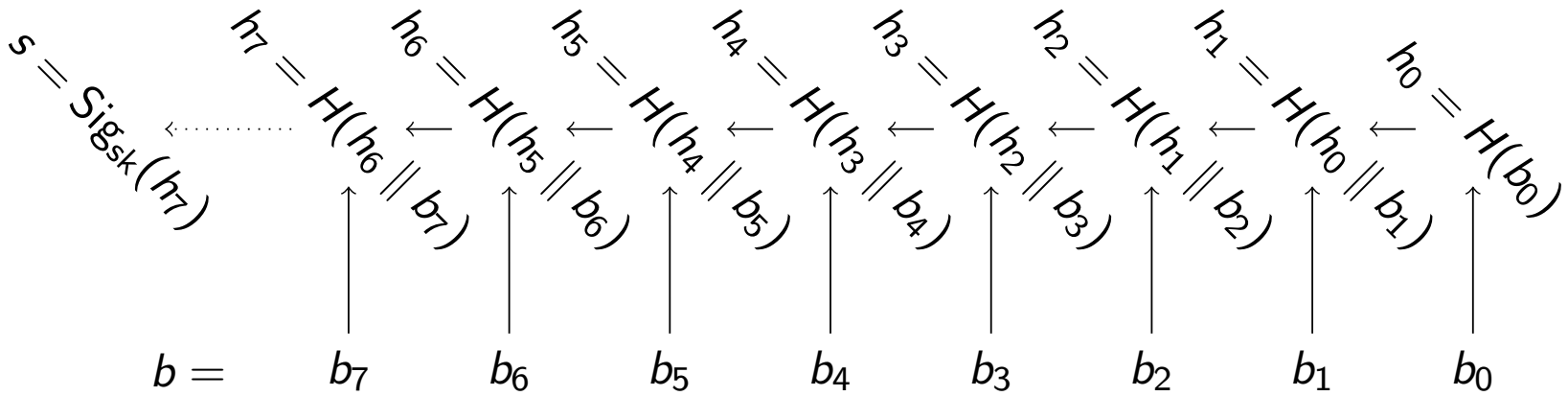


Construction: 1 Byte

Observation: Quantization with 2^n is truncation

$$\frac{b_7\, b_6\, b_5\, b_4\, b_3\, b_2\, b_1\, b_0}{2^3} \rightsquigarrow b_7\, b_6\, b_5\, b_4\, b_3\, x x x$$

Idea: Provide something instead of xxx?

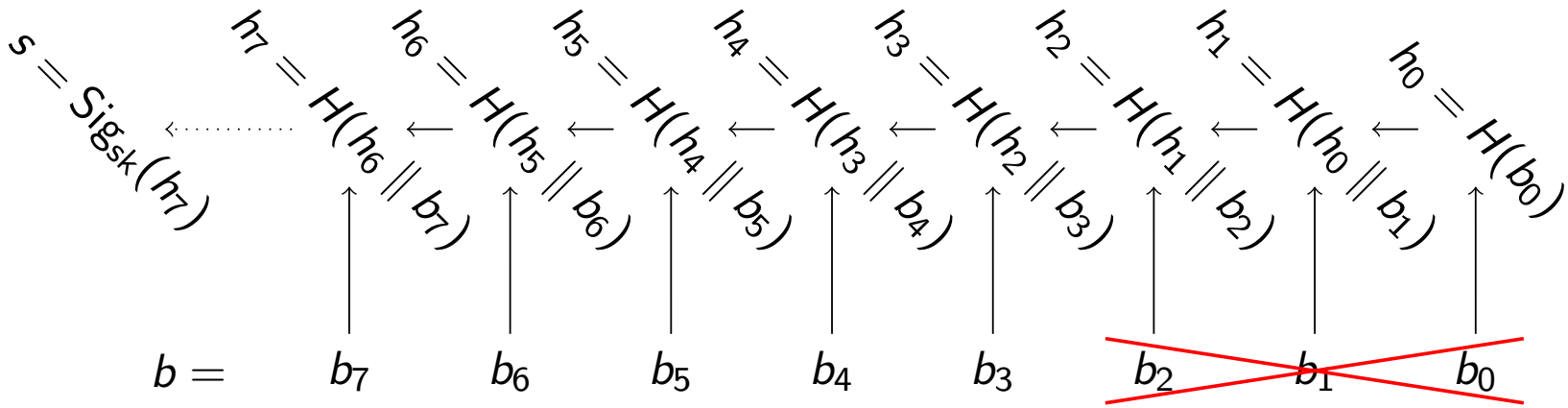


Construction: 1 Byte

Observation: Quantization with 2^n is truncation

$$\frac{b_7\, b_6\, b_5\, b_4\, b_3\, b_2\, b_1\, b_0}{2^3} \rightsquigarrow b_7\, b_6\, b_5\, b_4\, b_3\, x x x$$

Idea: Provide something instead of xxx?

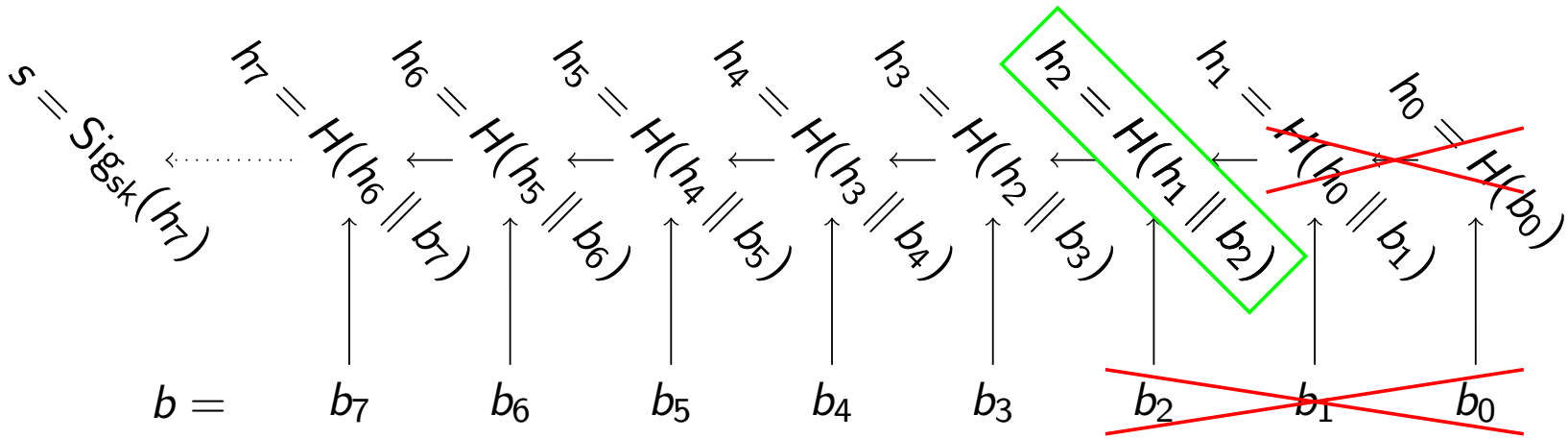


Construction: 1 Byte

Observation: Quantization with 2^n is truncation

$$\frac{b_7\, b_6\, b_5\, b_4\, b_3\, b_2\, b_1\, b_0}{2^3} \rightsquigarrow b_7\, b_6\, b_5\, b_4\, b_3\, x x x$$

Idea: Provide something instead of xxx?



However...

... it is super inefficient!

Construction: Images

But...

- ...each of the 64 DCT coefficients are truncated the same in every 8×8 block
- and images generally have many pixels/blocks

Construction: Images

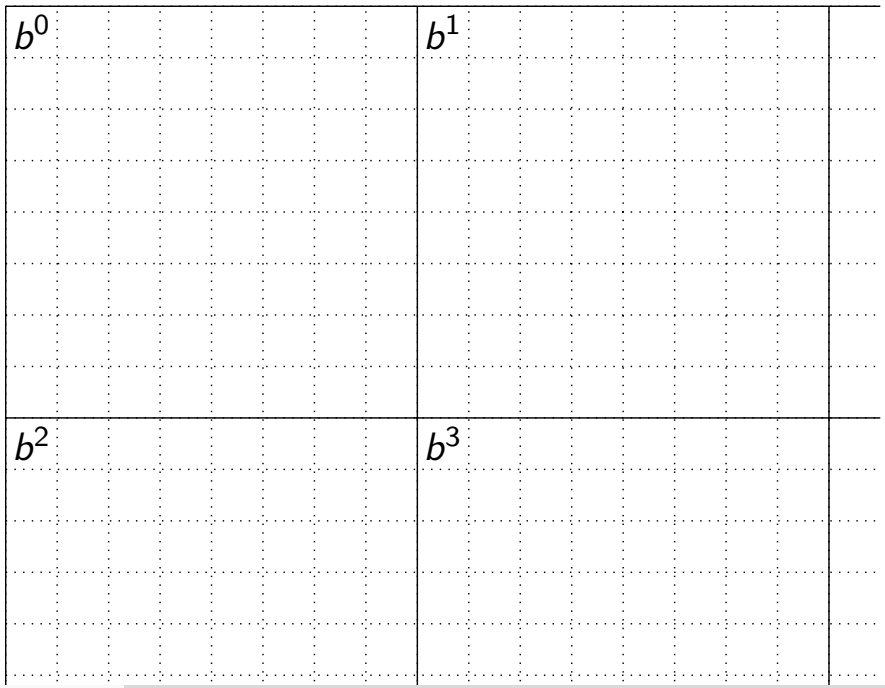
But...

- ...each of the 64 DCT coefficients are truncated the same in every 8×8 block
- and images generally have many pixels/blocks

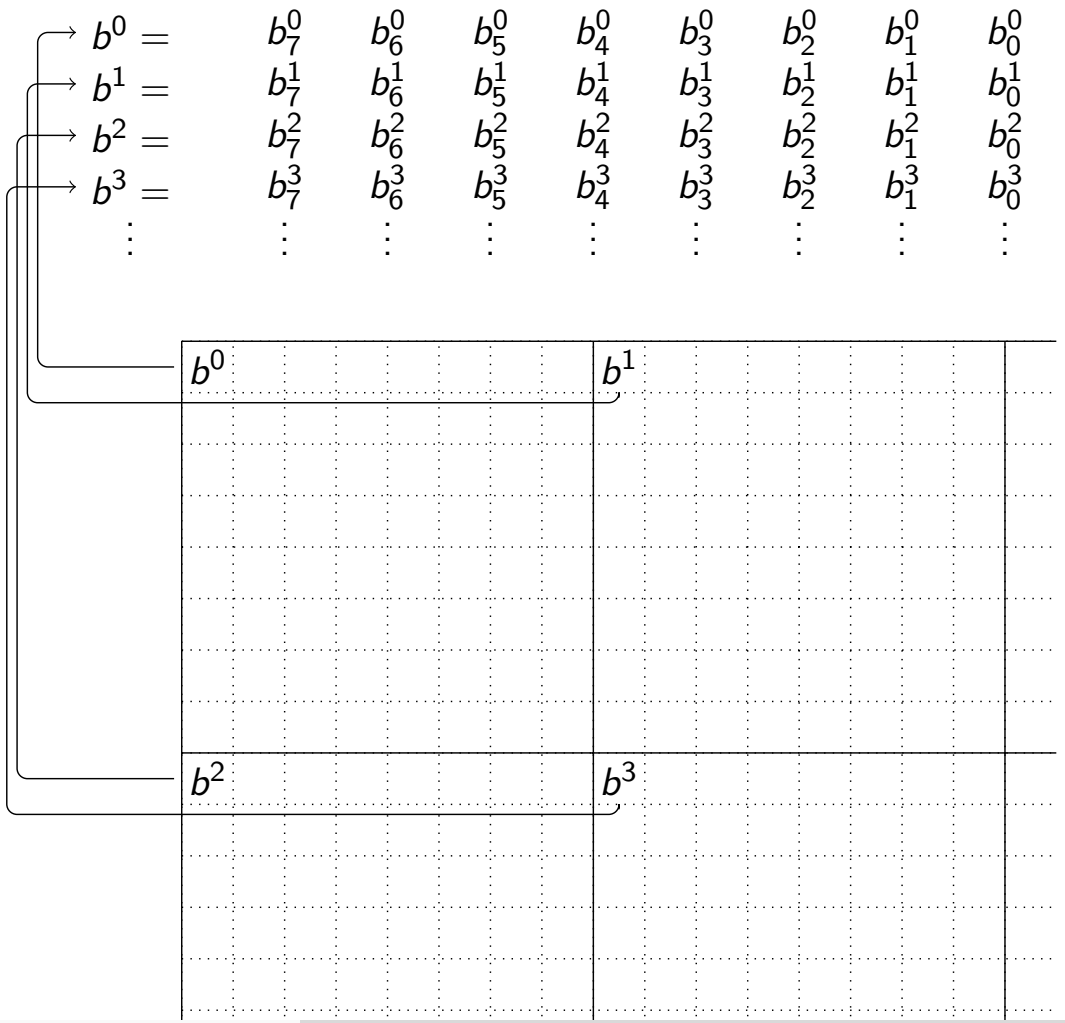
Our solution

- Generate signature by "combining" matching DCT coefficients and then generate hash chains and hash ends together.
- Compress using quantization table with powers of two and update signature to include relevant nodes.

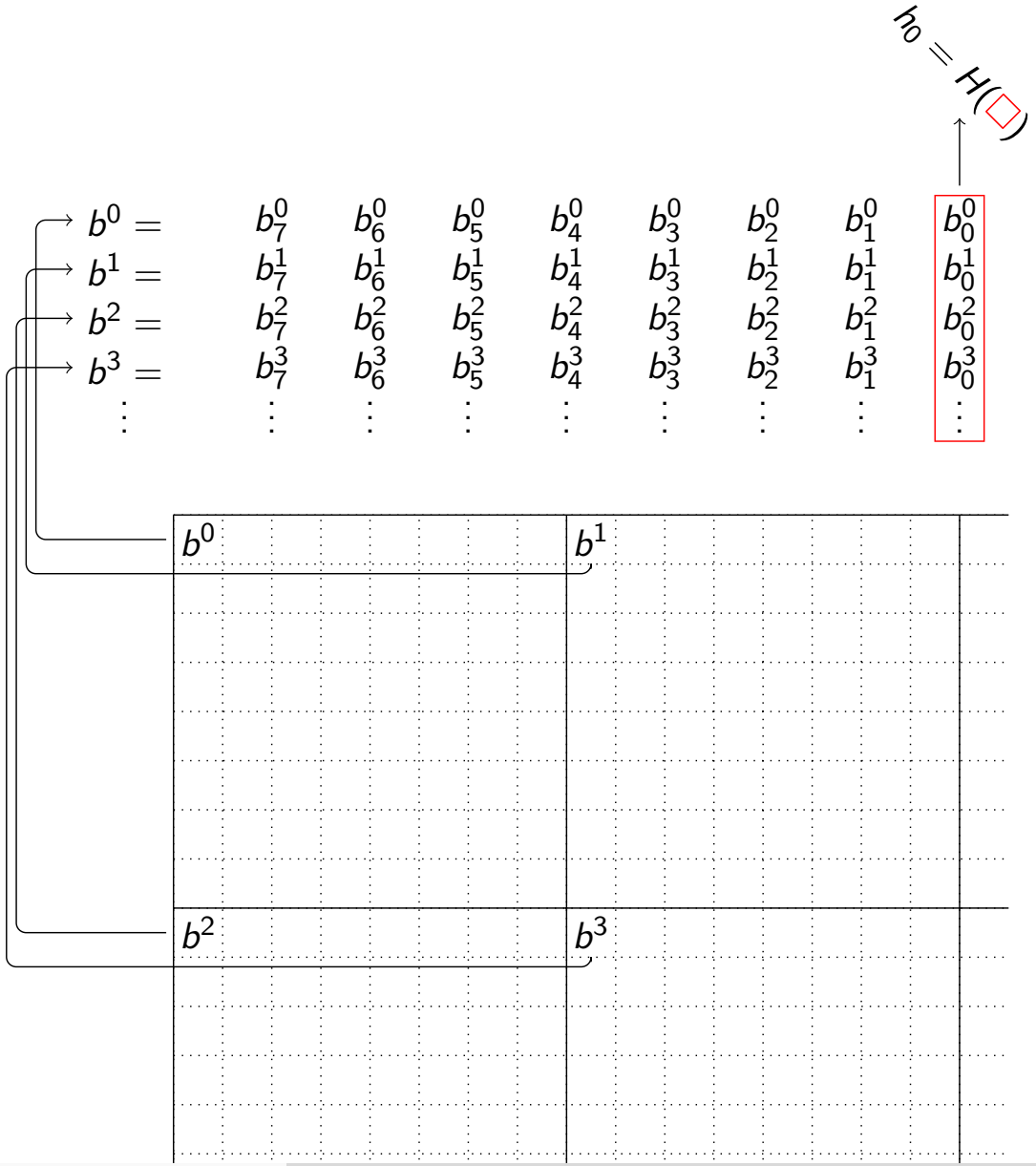
Construction: Images



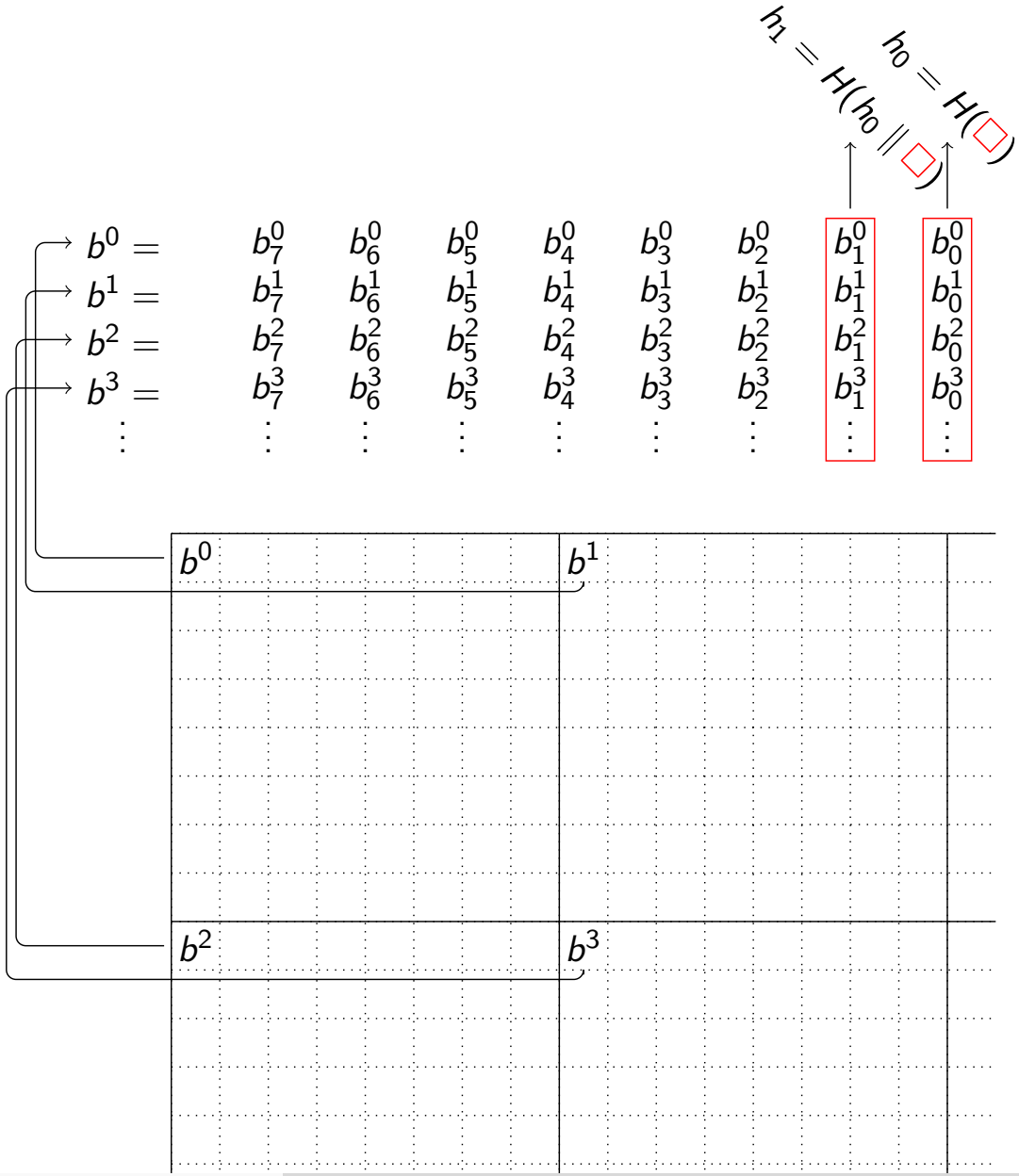
Construction: Images



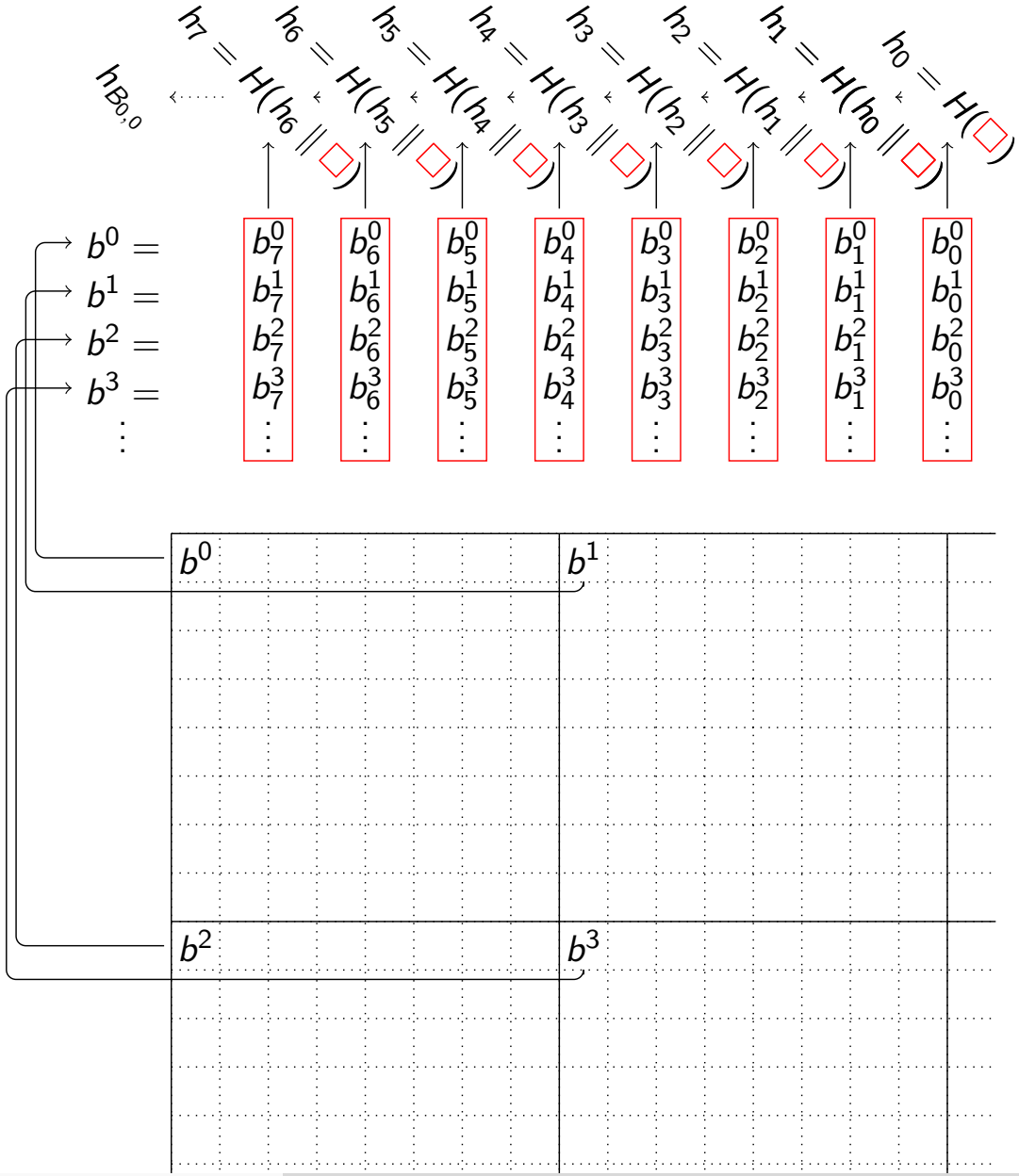
Construction: Images



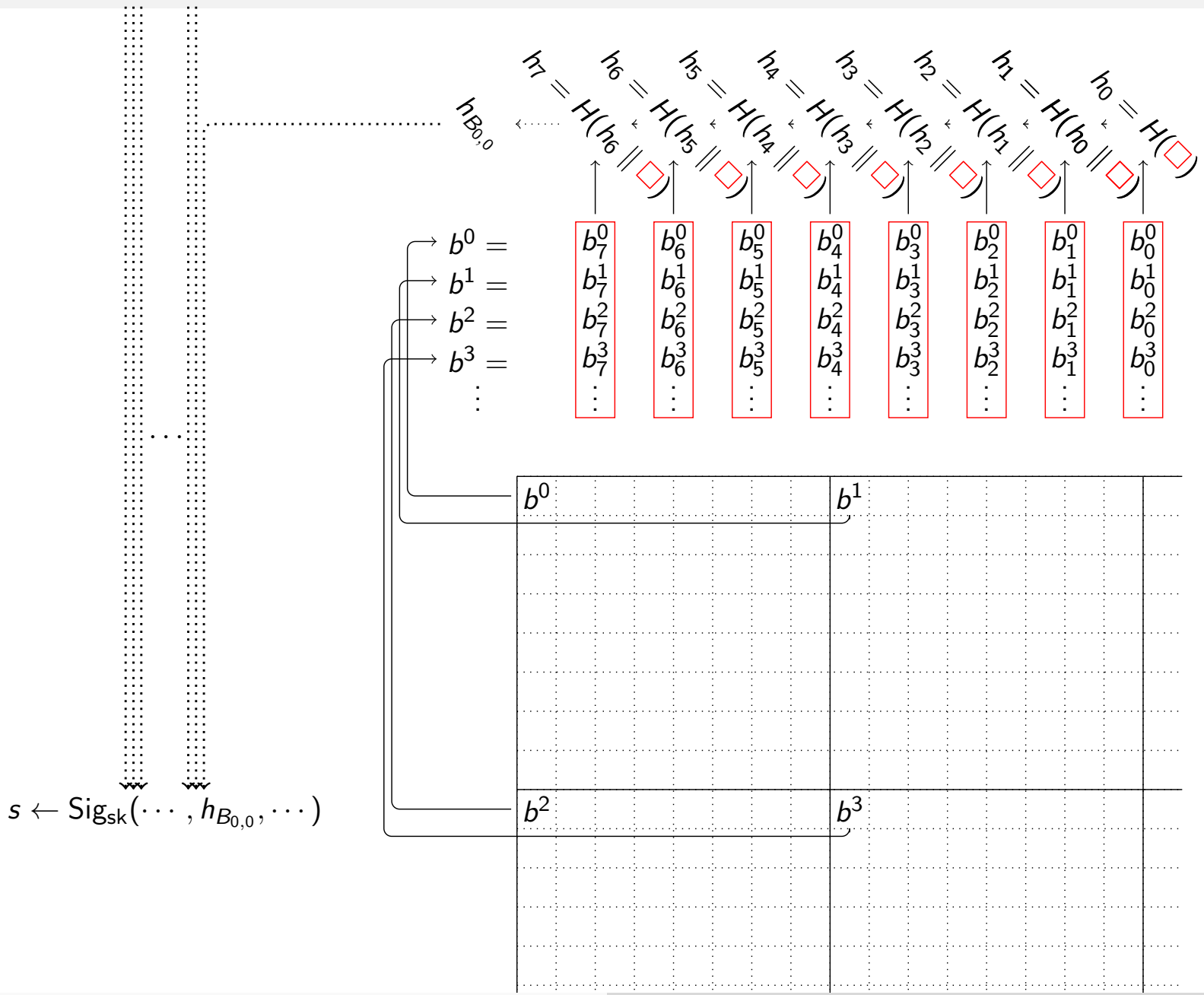
Construction: Images



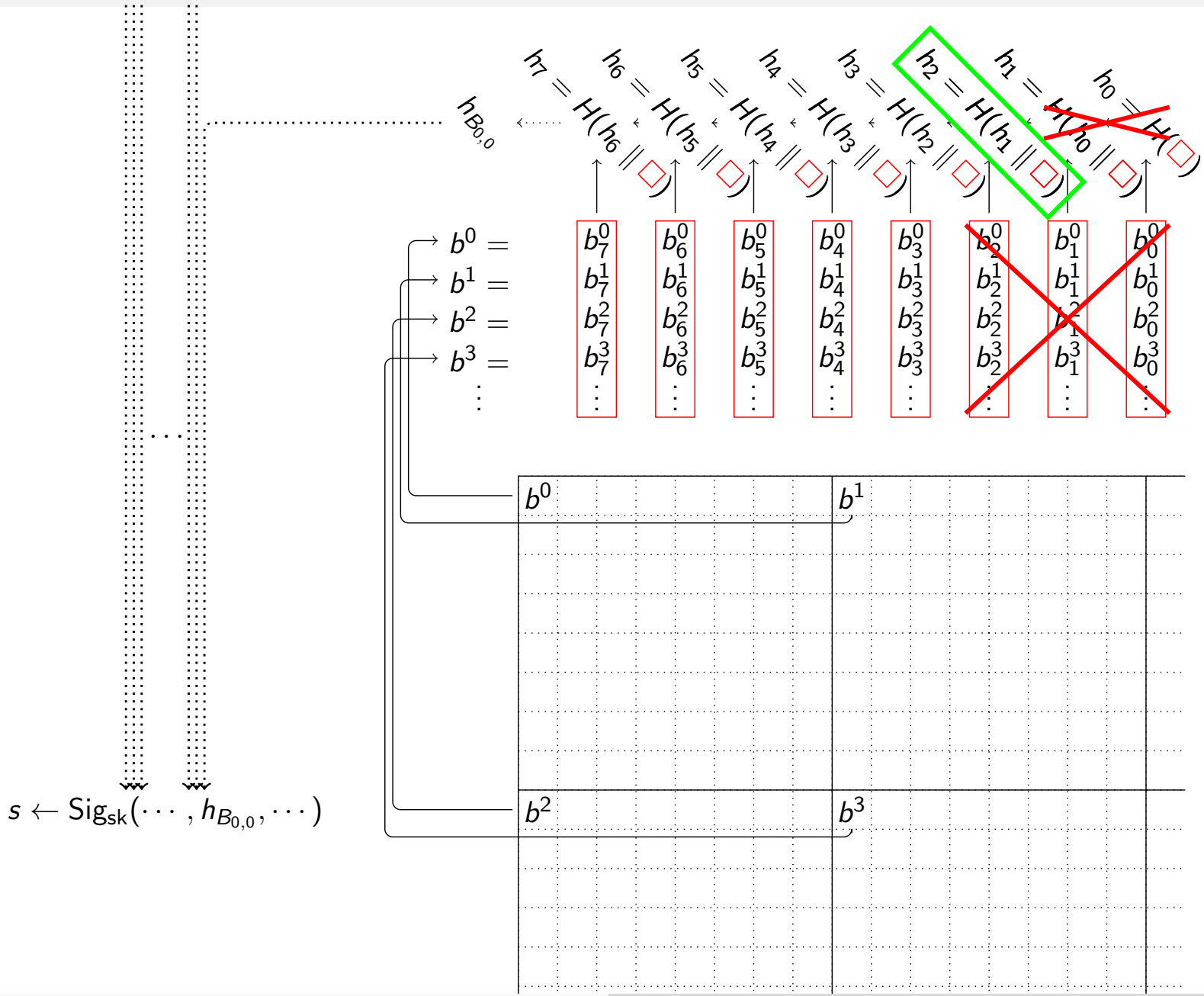
Construction: Images



Construction: Images



Construction: Images



Construction: Summary

Digital signature scheme allowing image compression

Construction: Summary

Digital signature scheme allowing image compression

Let H be a cryptographic hash function and
$\mathsf{DS} = (\mathsf{KeyGen}^{\mathsf{DS}}, \mathsf{Sign}^{\mathsf{DS}}, \mathsf{Verify}^{\mathsf{DS}})$ a standard digital signature scheme.

Construction: Summary

Digital signature scheme allowing image compression

Let H be a cryptographic hash function and $\mathsf{DS} = (\mathsf{KeyGen}^{\mathsf{DS}}, \mathsf{Sign}^{\mathsf{DS}}, \mathsf{Verify}^{\mathsf{DS}})$ a standard digital signature scheme.

- $\mathsf{sk}, \mathsf{pk} \leftarrow \mathsf{KeyGen}(1^\lambda)$: Identical to $\mathsf{KeyGen}^{\mathsf{DS}}(1^\lambda)$.

Construction: Summary

Digital signature scheme allowing image compression

Let H be a cryptographic hash function and $\mathsf{DS} = (\mathsf{KeyGen}^{\mathsf{DS}}, \mathsf{Sign}^{\mathsf{DS}}, \mathsf{Verify}^{\mathsf{DS}})$ a standard digital signature scheme.

- $\mathsf{sk}, \mathsf{pk} \leftarrow \mathsf{KeyGen}(1^\lambda)$: Identical to $\mathsf{KeyGen}^{\mathsf{DS}}(1^\lambda)$.
- $s \leftarrow \mathsf{Sign}_{\mathsf{sk}}(i)$: Compute the 128 chains of hashes, sign the ends using $\mathsf{Sign}^{\mathsf{DS}}$.

Construction: Summary

Digital signature scheme allowing image compression

Let H be a cryptographic hash function and $\mathsf{DS} = (\mathsf{KeyGen}^{\mathsf{DS}}, \mathsf{Sign}^{\mathsf{DS}}, \mathsf{Verify}^{\mathsf{DS}})$ a standard digital signature scheme.

- $\mathsf{sk}, \mathsf{pk} \leftarrow \mathsf{KeyGen}(1^\lambda)$: Identical to $\mathsf{KeyGen}^{\mathsf{DS}}(1^\lambda)$.
- $s \leftarrow \mathsf{Sign}_{\mathsf{sk}}(i)$: Compute the 128 chains of hashes, sign the ends using $\mathsf{Sign}^{\mathsf{DS}}$.
- $i', s' \leftarrow \mathsf{Compress}(i, s, P)$: Compress i using quantization tables P, compute chains of hashes and extract relevant ones. Add these to s.

Construction: Summary

Digital signature scheme allowing image compression

Let H be a cryptographic hash function and $\mathrm{DS} = (\mathrm{KeyGen}^{\mathrm{DS}}, \mathrm{Sign}^{\mathrm{DS}}, \mathrm{Verify}^{\mathrm{DS}})$ a standard digital signature scheme.

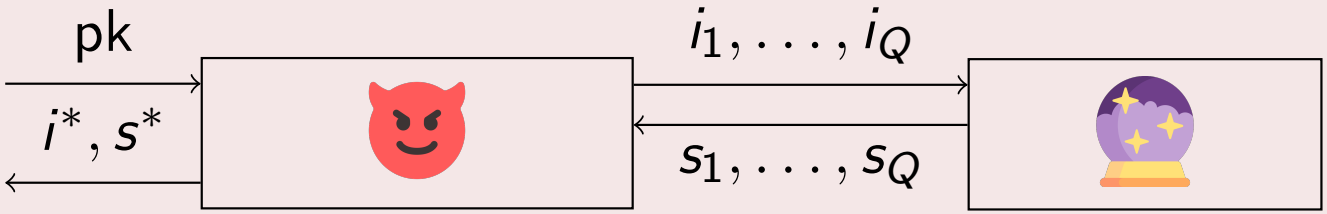
- $\mathrm{sk}, \mathrm{pk} \leftarrow \mathrm{KeyGen}(1^\lambda)$: Identical to $\mathrm{KeyGen}^{\mathrm{DS}}(1^\lambda)$.
- $s \leftarrow \mathrm{Sign}_{\mathrm{sk}}(i)$: Compute the 128 chains of hashes, sign the ends using $\mathrm{Sign}^{\mathrm{DS}}$.
- $i', s' \leftarrow \mathrm{Compress}(i, s, P)$: Compress i using quantization tables P, compute chains of hashes and extract relevant ones. Add these to s.
- $0/1 \leftarrow \mathrm{Verify}_{\mathrm{pk}}(i, s)$: Use i and hashes in s, if any, to find chain ends. Verify with $\mathrm{Verify}^{\mathrm{DS}}$.

Security Notion

Unforgeability

Unforgeability Game

$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}()$



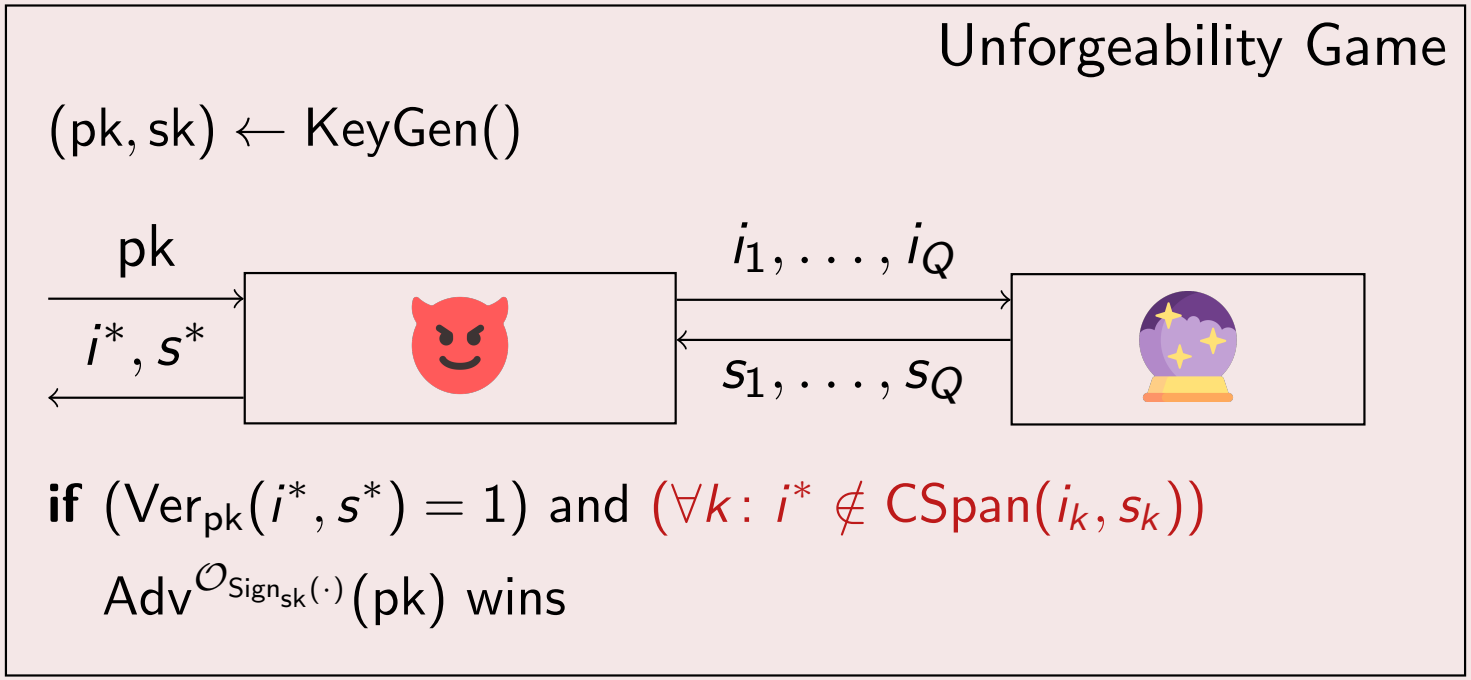
if $(\mathsf{Ver}_{\mathsf{pk}}(i^*, s^*) = 1)$ and $(\forall k : i^* \notin \mathsf{CSpan}(i_k, s_k))$

$\mathsf{Adv}^{\mathcal{O}_{\mathsf{Sign}_{\mathsf{sk}}(\cdot)}}(\mathsf{pk})$ wins

$\mathsf{CSpan}(i, s)$: all images i can be compressed to.

Security Notion

Unforgeability



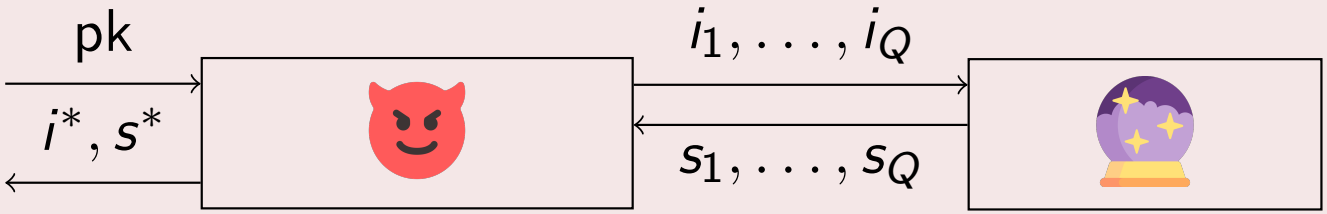
$\mathsf{CSpan}(i, s)$: all images i can be compressed to.

Security Notion

Unforgeability

Unforgeability Game

$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}()$



if $(\mathsf{Ver}_{\mathsf{pk}}(i^*, s^*) = 1)$ and $(\forall k : i^* \notin \mathsf{CSpan}(i_k, s_k))$

$\mathsf{Adv}^{\mathcal{O}_{\mathsf{Sign}_{\mathsf{sk}}(\cdot)}}(\mathsf{pk})$ wins

$\mathsf{CSpan}(i, s)$: all images i can be compressed to.

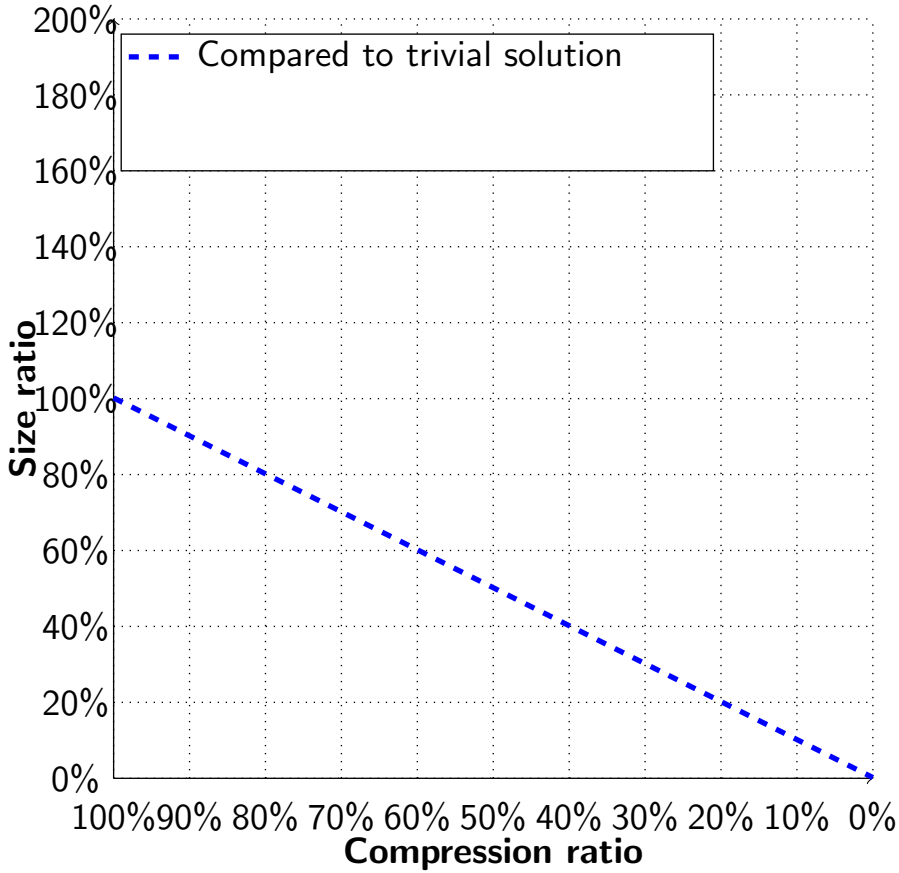
Claim

Our scheme signature scheme allowing image compression is unforgeable.

Performance: Signature Size

Uncompressed image 2 megabytes

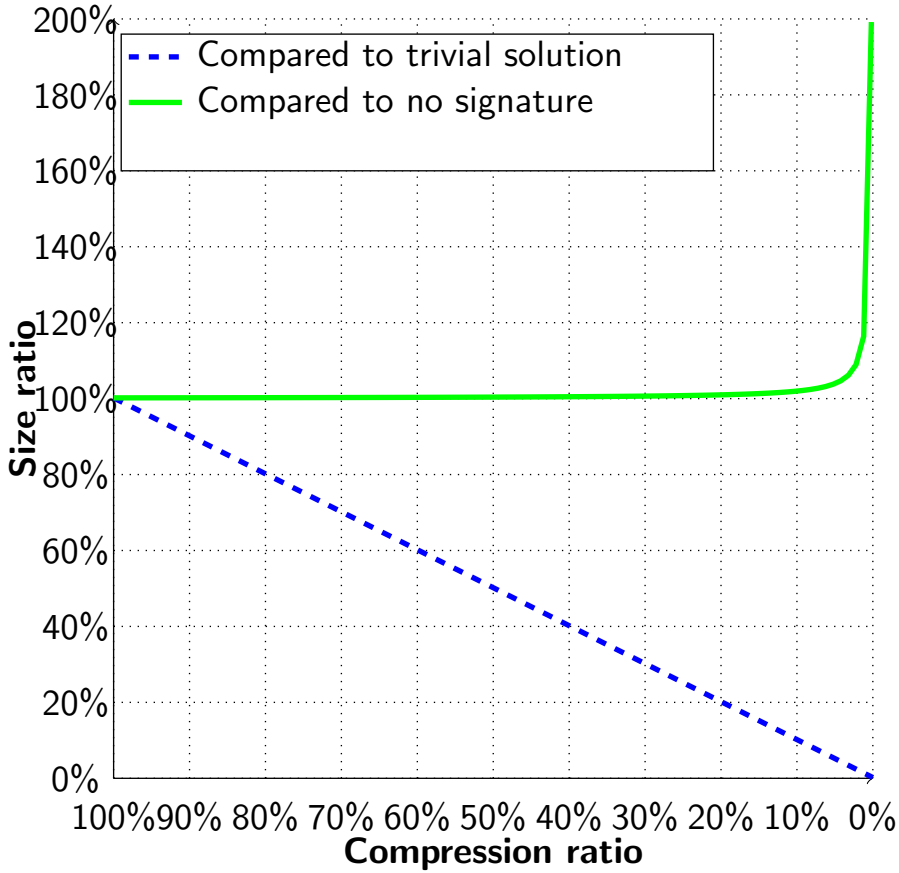
Performance: Signature Size



Uncompressed image 2 megabytes

- - - $|S| = 512$ bits.

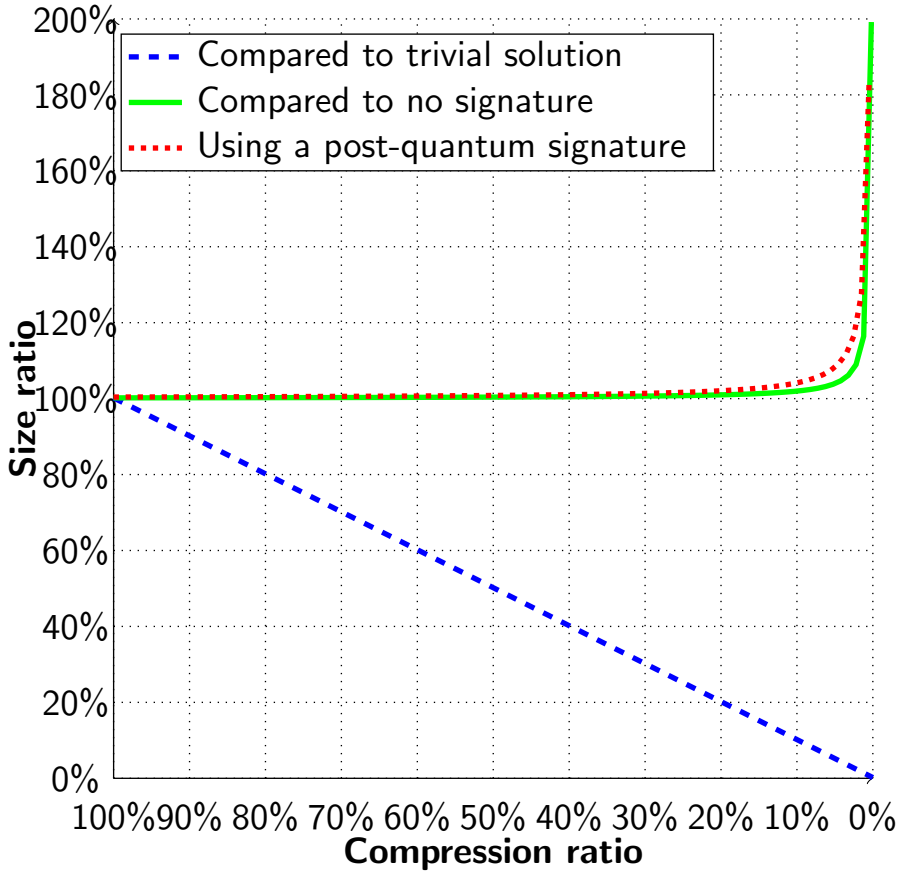
Performance: Signature Size



Uncompressed image 2 megabytes

- $|S| = 512$ bits.
- $|S| = 512$ bits, $|H| = 256$ bits.

Performance: Signature Size



Uncompressed image 2 megabytes

- $|S| = 512$ bits.
- $|S| = 512$ bits, $|H| = 256$ bits.
- $|S| = 36760$ bits, $|H| = 256$ bits.

Visual Evaluation: Ocular

Visual Evaluation: Ocular

Uncompressed

Classic (QF50)

Powers of two

Visual Evaluation: Numeric

	Size	MS-SSIM	FSIMc	MSE	PSNR
QF25					
QF50					
QF80					

Average over all images in [PLZ+09].

Visual Evaluation: Numeric

		Size	MS-SSIM	FSIMc	MSE	PSNR
QF25	**Our tables**	16.0 kB	0.960	0.978	77.526	29.749
	Unmodified	15.1 kB	0.959	0.978	78.900	29.655
	[JWL11] (64)	10.4 kB	0.914	0.936	109.016	28.021
	[JWL11] (32)	20.5 kB	0.961	0.976	46.071	31.706
QF50	**Our tables**	25.4 kB	0.979	0.991	45.831	32.008
	Unmodified	24.4 kB	0.979	0.991	45.910	31.988
	[JWL11] (32)	20.5 kB	0.961	0.976	46.071	31.706
	[JWL11] (16)	36.5 kB	0.983	0.992	18.451	35.605
QF80	**Our tables**	43.9 kB	0.990	0.997	20.256	35.402
	Unmodified	43.4 kB	0.991	0.997	20.432	35.364
	[JWL11] (16)	36.5 kB	0.983	0.992	18.451	35.605
	[JWL11] (8)	60.4 kB	0.993	0.997	7.532	39.439

Average over all images in [PLZ+09].

Visual Evaluation: Numeric

		Size	MS-SSIM	FSIMc	MSE	PSNR
QF25	**Our tables**	16.0 kB	0.960	0.978	77.526	29.749
	Unmodified	15.1 kB	0.959	0.978	78.900	29.655
	[JWL11] (64)	10.4 kB	0.914	0.936	109.016	28.021
	[JWL11] (32)	20.5 kB	0.961	0.976	46.071	31.706
QF50	**Our tables**	25.4 kB	0.979	0.991	45.831	32.008
	Unmodified	24.4 kB	0.979	0.991	45.910	31.988
	[JWL11] (32)	20.5 kB	0.961	0.976	46.071	31.706
	[JWL11] (16)	36.5 kB	0.983	0.992	18.451	35.605
QF80	**Our tables**	43.9 kB	0.990	0.997	20.256	35.402
	Unmodified	43.4 kB	0.991	0.997	20.432	35.364
	[JWL11] (16)	36.5 kB	0.983	0.992	18.451	35.605
	[JWL11] (8)	60.4 kB	0.993	0.997	7.532	39.439

Average over all images in [PLZ+09].

Final Remarks

Summary

Digital signature scheme for images allowing some compression.

- Efficient and secure wrt. a natural notion of unforgeability.
- Visually very similar to unrestricted JPEG compression.
- Fully backwards compatible with JPEG viewers.

Final Remarks

Summary

Digital signature scheme for images allowing some compression.

- Efficient and secure wrt. a natural notion of unforgeability.
- Visually very similar to unrestricted JPEG compression.
- Fully backwards compatible with JPEG viewers.

Future work

- Standardize quantization tables.
- For our use case:
  - Future design of full scheme (back links, retractions?)
  - Collaboration with social media and news organisations.
  - Other image operations? Other media types?

Thank you for listening.

ePrint 2024/588

[Erf24]

References

[EH24] Kayla Epstein and Madeline Halpert. "'The air in the courtroom turned to stone'". In: *the BBC* (May 31, 2024). URL: https://www.bbc.com/news/articles/c2xx1dyd70xo (visited on 06/19/2024).

[Erf24] Simon Erfurth. "Digital Signatures for Authenticating Compressed JPEG Images". In: *IACR Cryptology ePrint Arch.* (2024), p. 588. URL: https://eprint.iacr.org/2024/588.

[JWL11] Rob Johnson, Leif Walsh, and Michael Lamb. "Homomorphic Signatures for Digital Photographs". In: *Financial Cryptography and Data Security - FC 2011*. Vol. 7035. LNCS. Springer, 2011, pp. 141–157. DOI: 10.1007/978-3-642-27576-0_12.

[PLZ+09] Nikolay Ponomarenko, Vladimir Lukin, Alexander Zelensky, Karen Egiazarian, Marco Carli,

Performance: Computation

	Computation Time	Signature Size

Performance: Computation

	Computation Time	**Signature Size**
Key generation	Same as $\text{KeyGen}^{\text{DS}}$	—

Performance: Computation

	Computation Time	**Signature Size**
Key generation	Same as $\text{KeyGen}^{\text{DS}}$	—
Signing	1025 hashes and time of Sign^{DS}	$\lvert S \rvert$

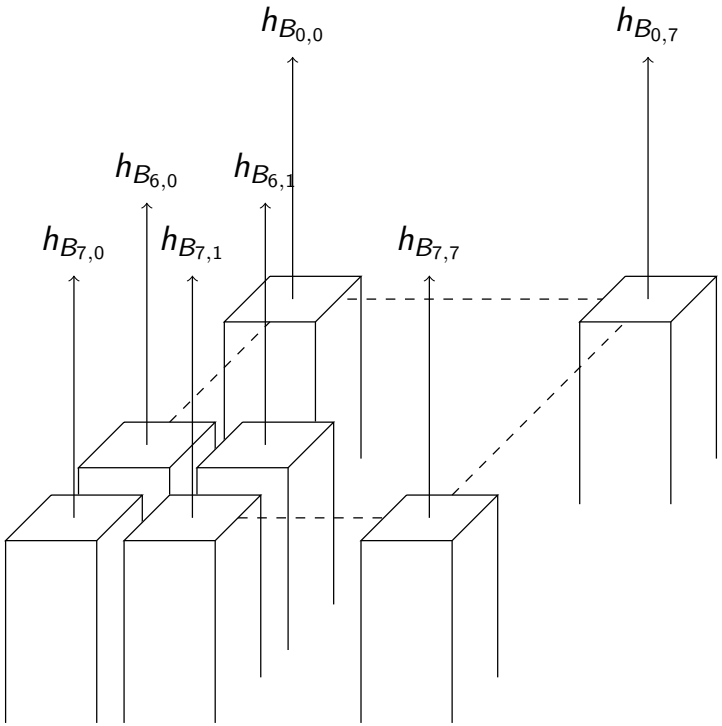
Performance: Computation

	Computation Time	**Signature Size**
Key generation	Same as $\text{KeyGen}^{\text{DS}}$	—
Signing	1025 hashes and time of Sign^{DS}	$\lvert S \rvert$
Compression	1025 hashes	$128\lvert H \rvert + \lvert S \rvert$

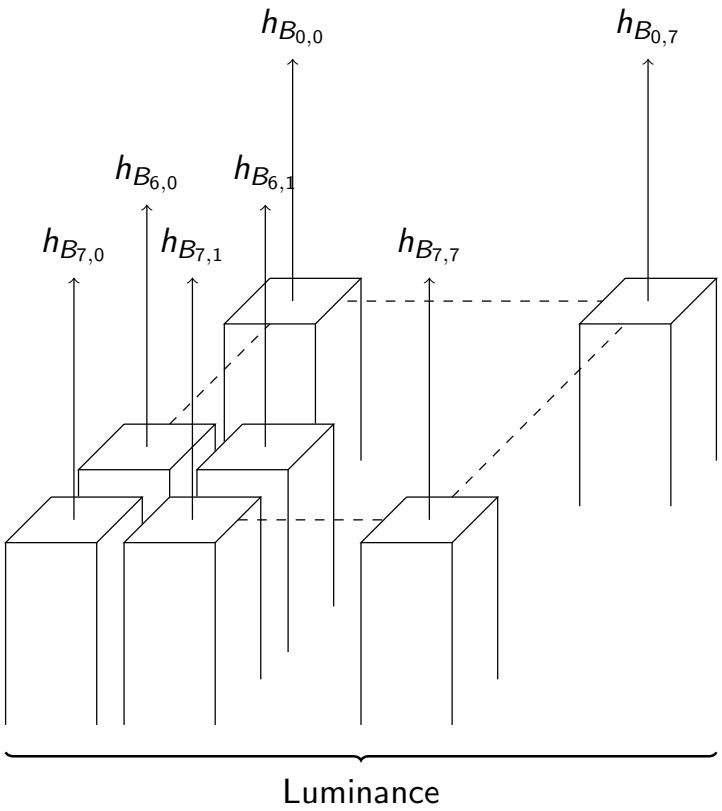
Performance: Computation

	Computation Time	**Signature Size**
Key generation	Same as $\text{KeyGen}^{\text{DS}}$	—
Signing	1025 hashes and time of Sign^{DS}	$\lvert S \rvert$
Compression	1025 hashes	$128\lvert H \rvert + \lvert S \rvert$
Verification	1025 hashes and time of $\text{Verify}^{\text{DS}}$	—

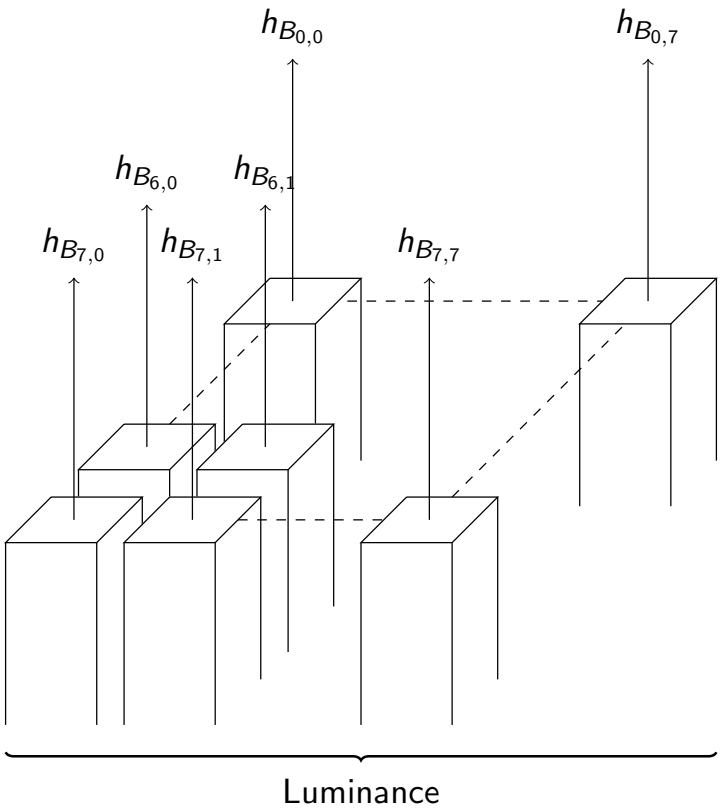
Construction: Images

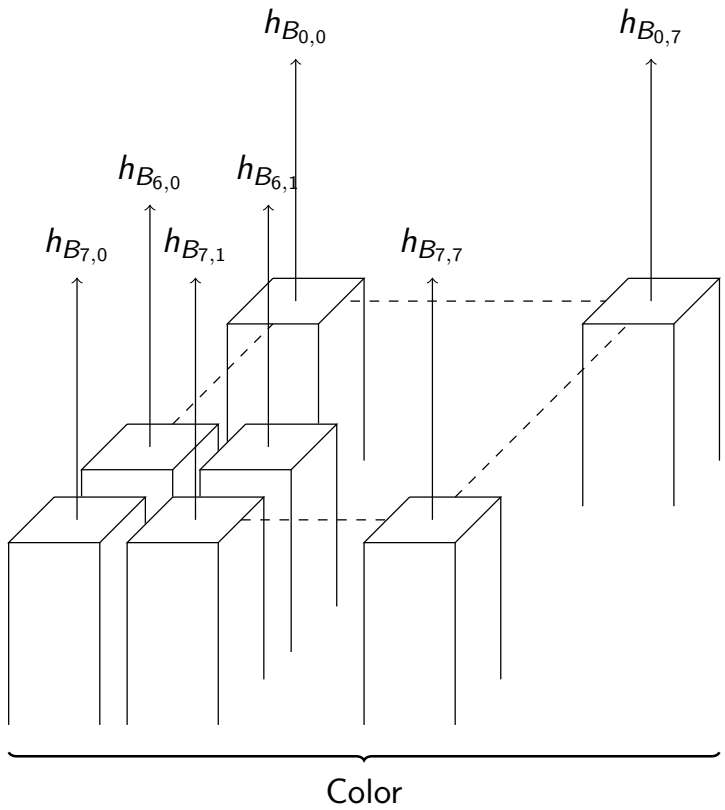


Construction: Images

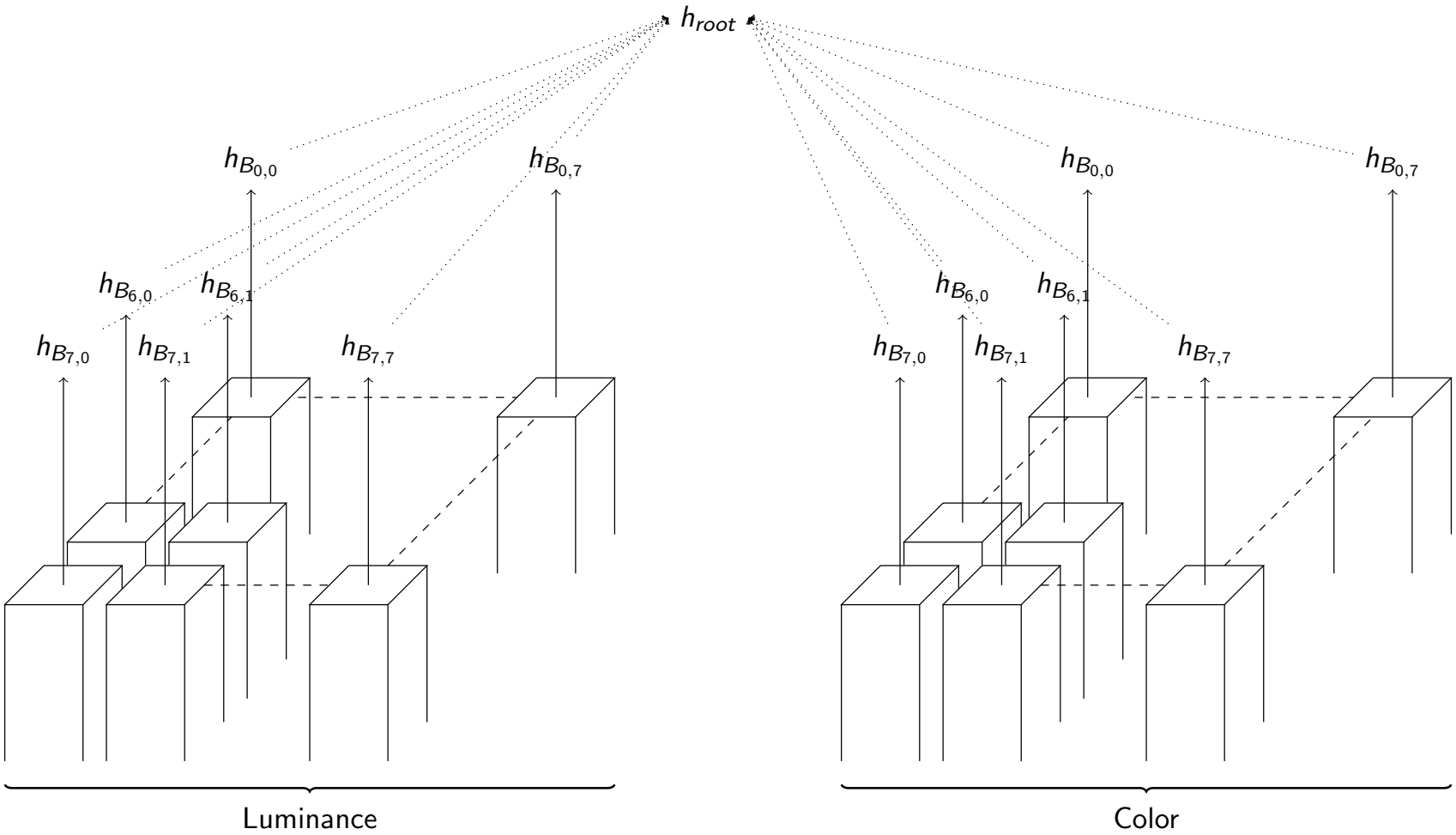


Construction: Images

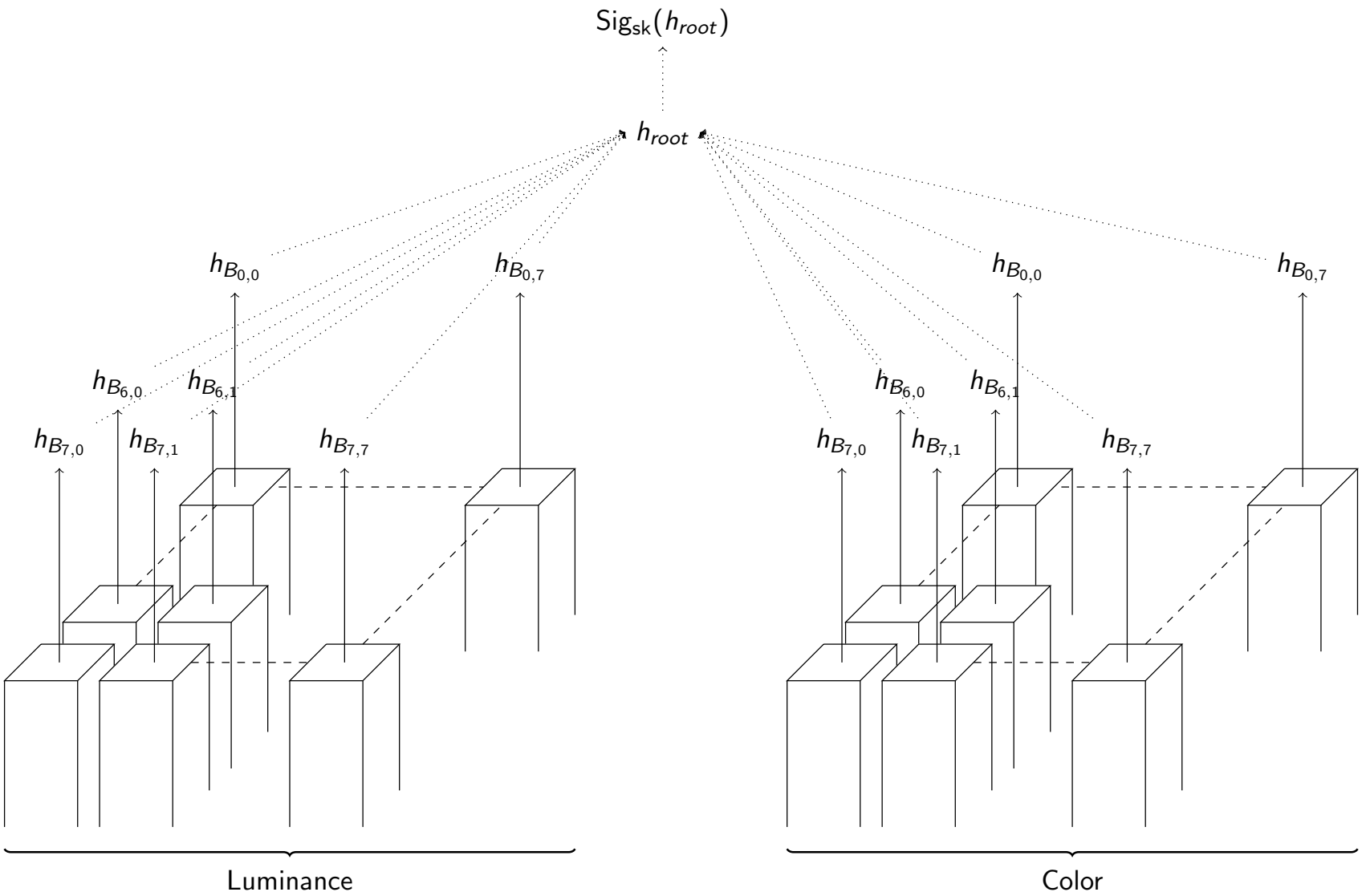




Construction: Images



Construction: Images



Proof of Security

Proof of Security

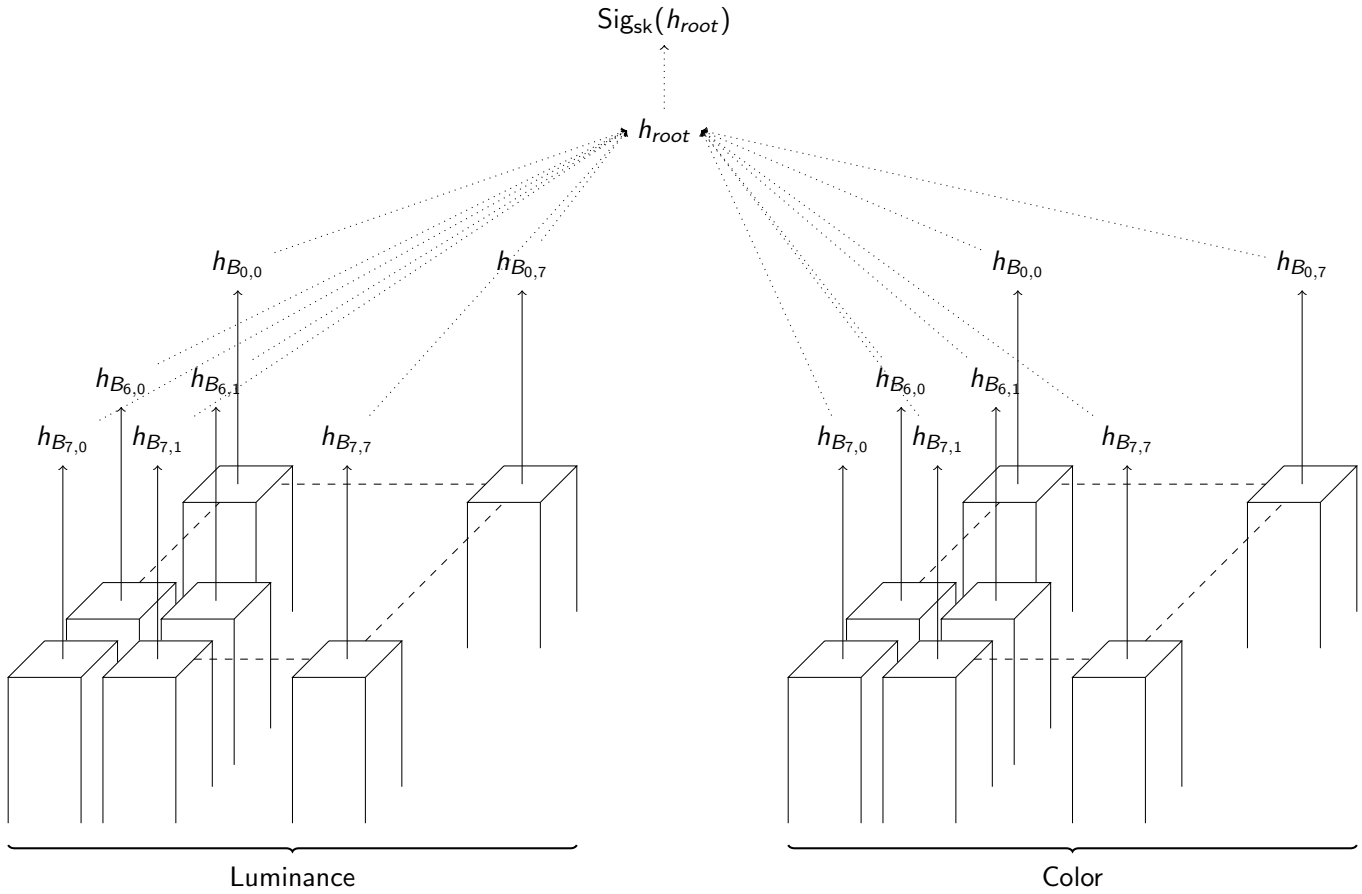
If $\forall k$: i^*'s h_{root} is different from i_k's h_{root}: Forgery against DS.

Proof of Security

Else let k be such that i^* and i_k have the same h_{root}.

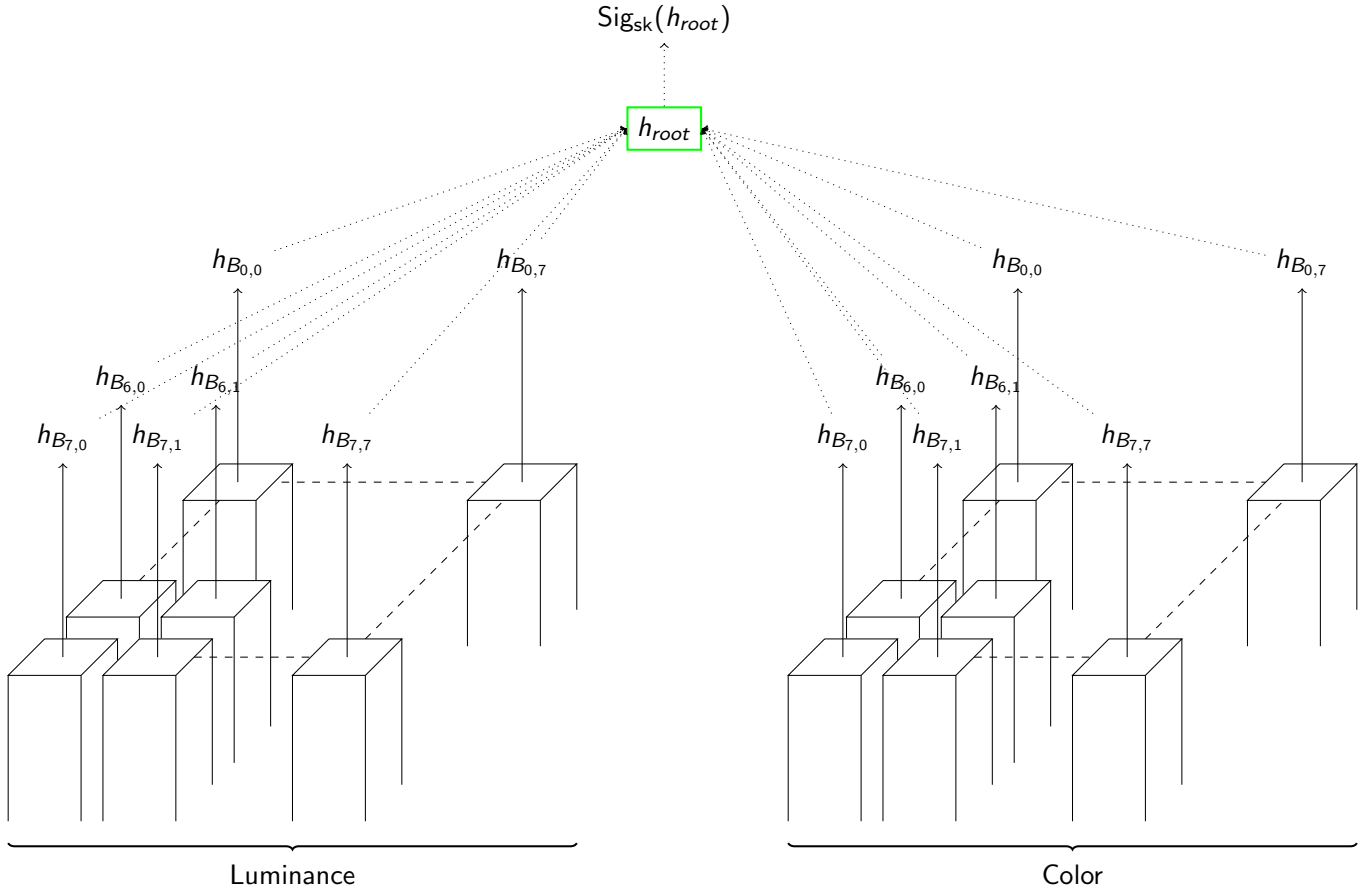
Proof of Security

Else let k be such that i^* and i_k have the same h_{root}.



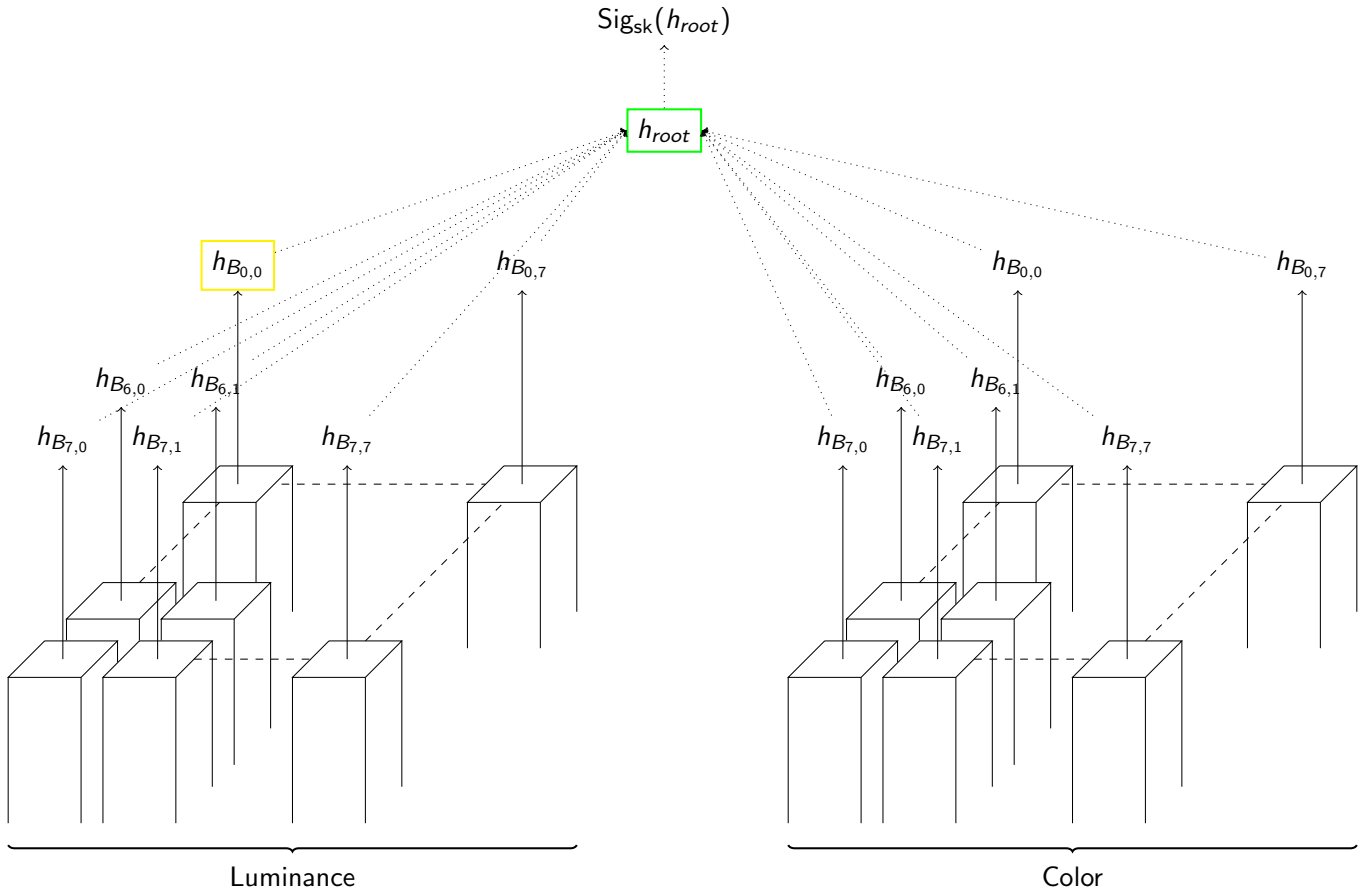
Proof of Security

Else let k be such that i^* and i_k have the same h_{root}.



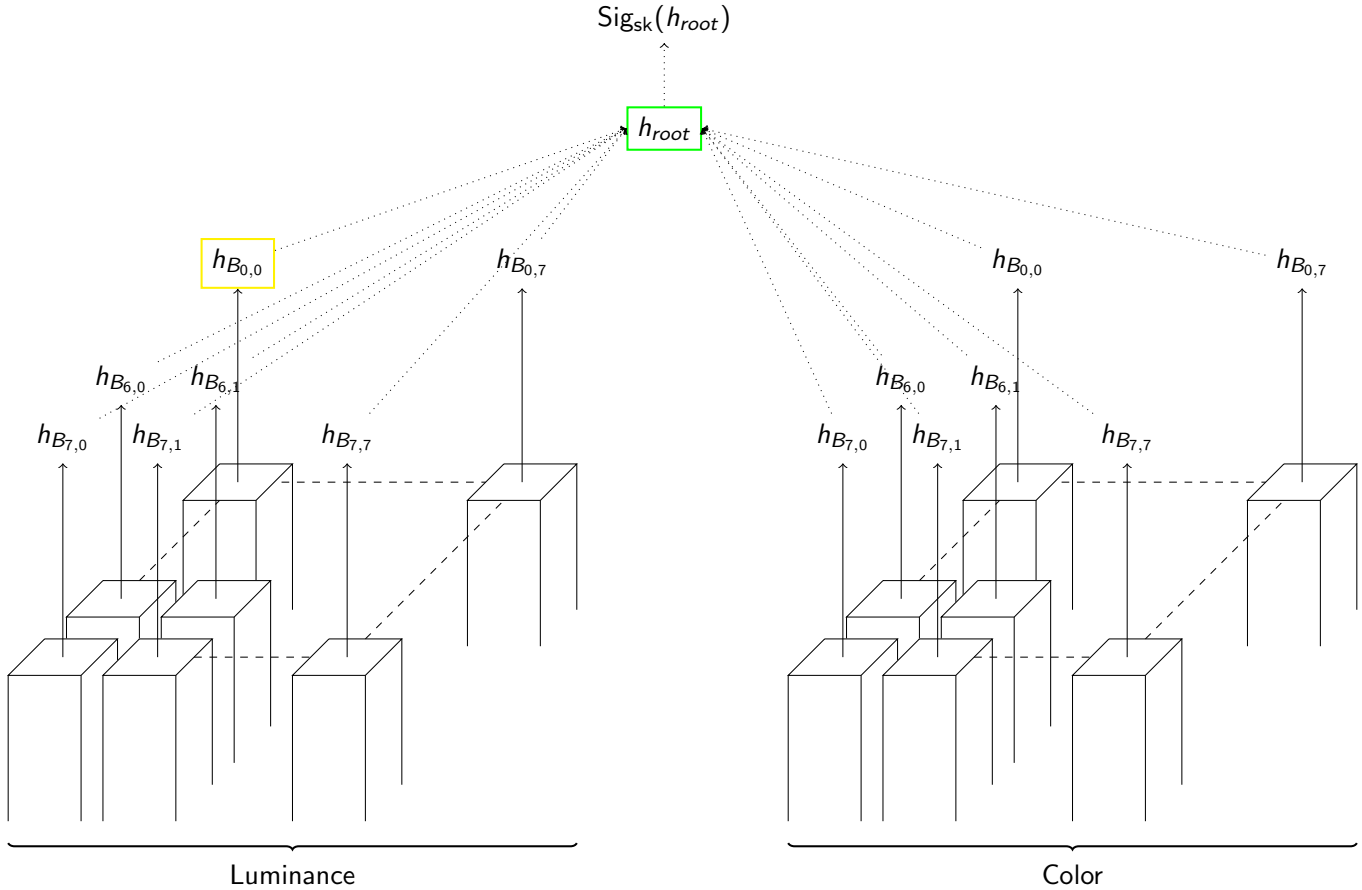
Proof of Security

Else let k be such that i^* and i_k have the same h_{root}.



Proof of Security

Else let k be such that i^* and i_k have the same h_{root}.

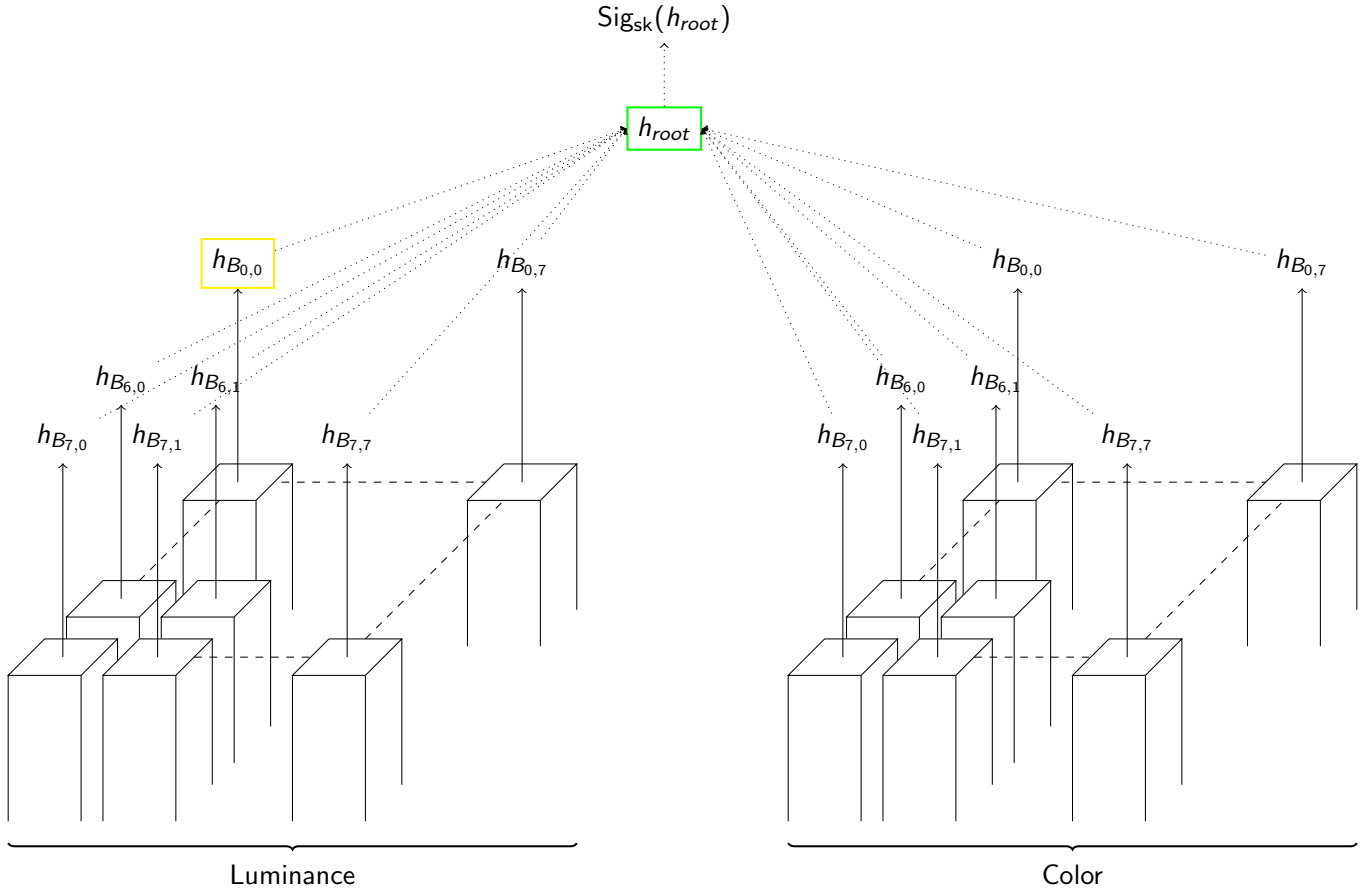


If $h_{B_{0,0}}$ is different for i^* and i^k: Collision for H:

$$H(\cdots, h^*_{B_{0,0}}, \cdots) = h_{root} = H(\cdots, h^k_{B_{0,0}}, \cdots).$$

Proof of Security

Else let k be such that i^* and i_k have the same h_{root}.



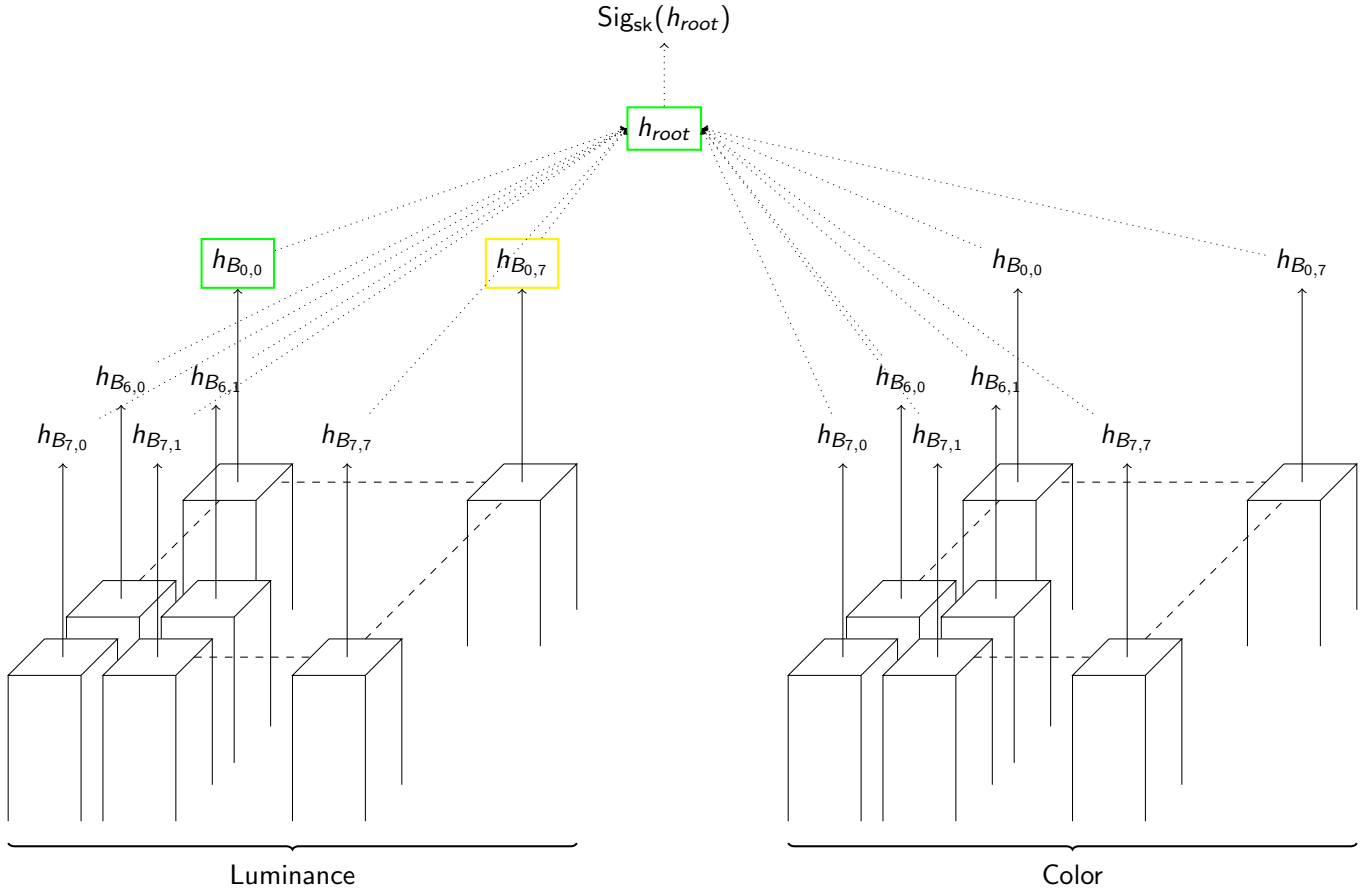
If $h_{B_{0,0}}$ is different for i^* and i^k: Collision for H:

$$H(\cdots, h^*_{B_{0,0}}, \cdots) = h_{root} = H(\cdots, h^k_{B_{0,0}}, \cdots).$$

Else move on.

Proof of Security

Else let k be such that i^* and i_k have the same h_{root}.



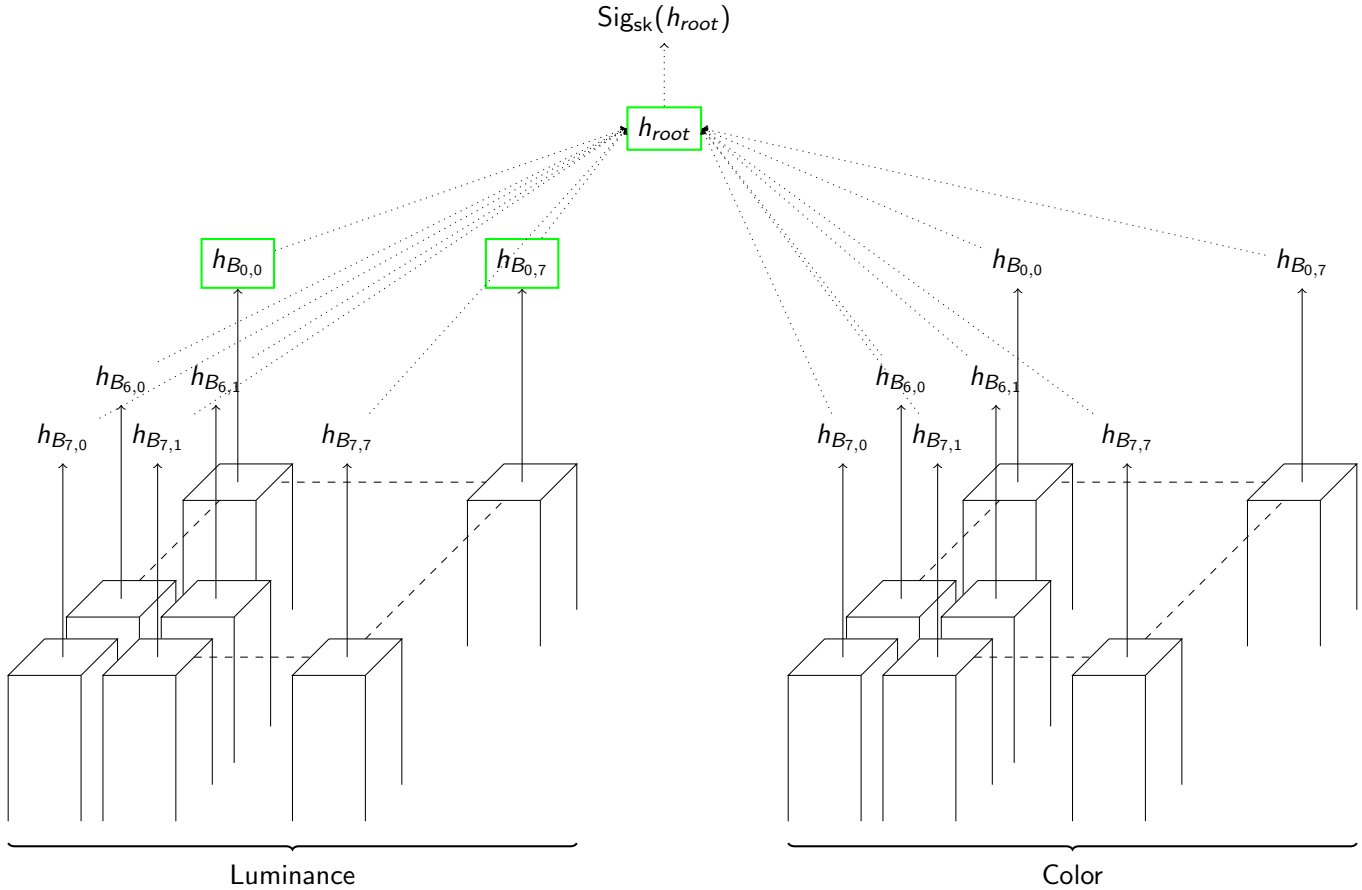
If $h_{B_{0,0}}$ is different for i^* and i^k: Collision for H:

$$H(\cdots, h^*_{B_{0,0}}, \cdots) = h_{root} = H(\cdots, h^k_{B_{0,0}}, \cdots).$$

Else move on.

Proof of Security

Else let k be such that i^* and i_k have the same h_{root}.



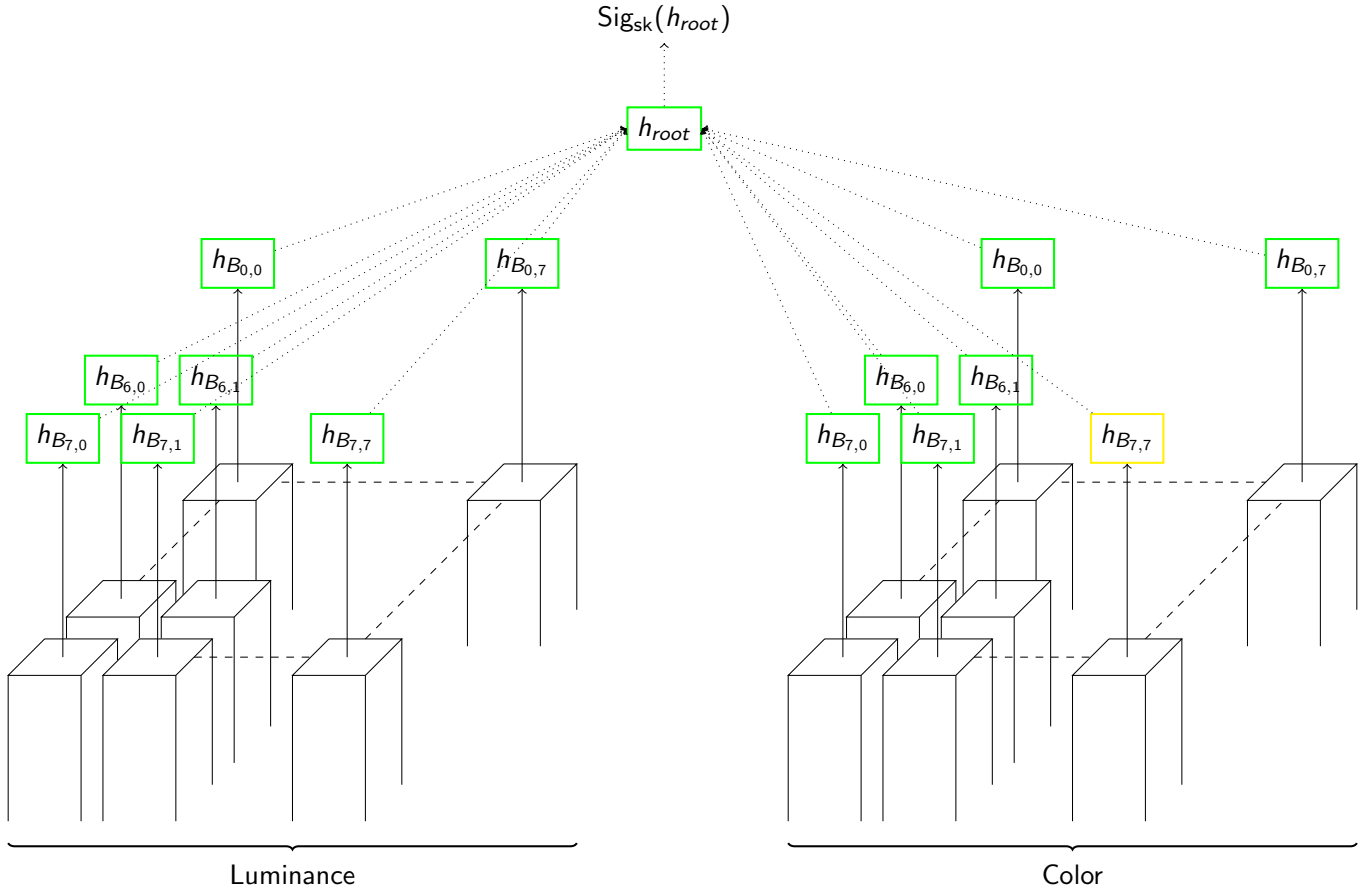
If $h_{B_{0,0}}$ is different for i^* and i^k: Collision for H:

$$H(\cdots, h^*_{B_{0,0}}, \cdots) = h_{root} = H(\cdots, h^k_{B_{0,0}}, \cdots).$$

Else move on.

Proof of Security

Else let k be such that i^* and i_k have the same h_{root}.



If $h_{B_{0,0}}$ is different for i^* and i^k: Collision for H:

$$H(\cdots, h^*_{B_{0,0}}, \cdots) = h_{root} = H(\cdots, h^k_{B_{0,0}}, \cdots).$$

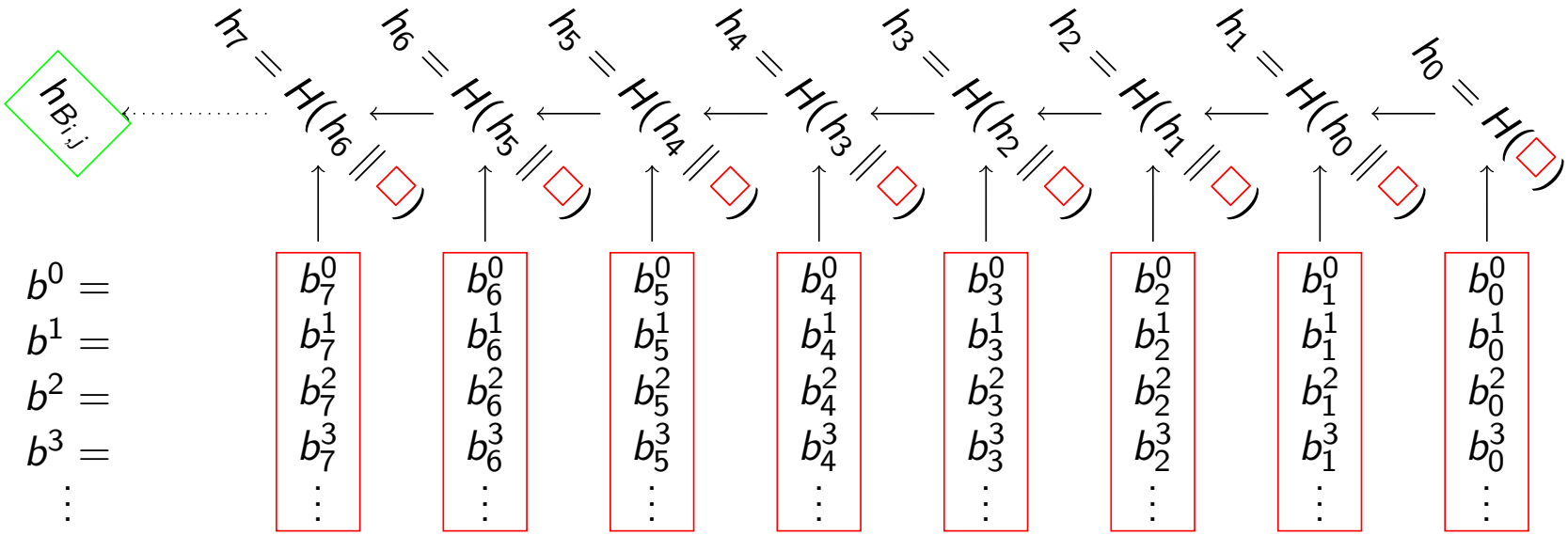
Else move on.

Proof of Security

Either we have found $h^*_{B_{i,j}} \neq h^k_{B_{i,j}}$ and a collision to H.

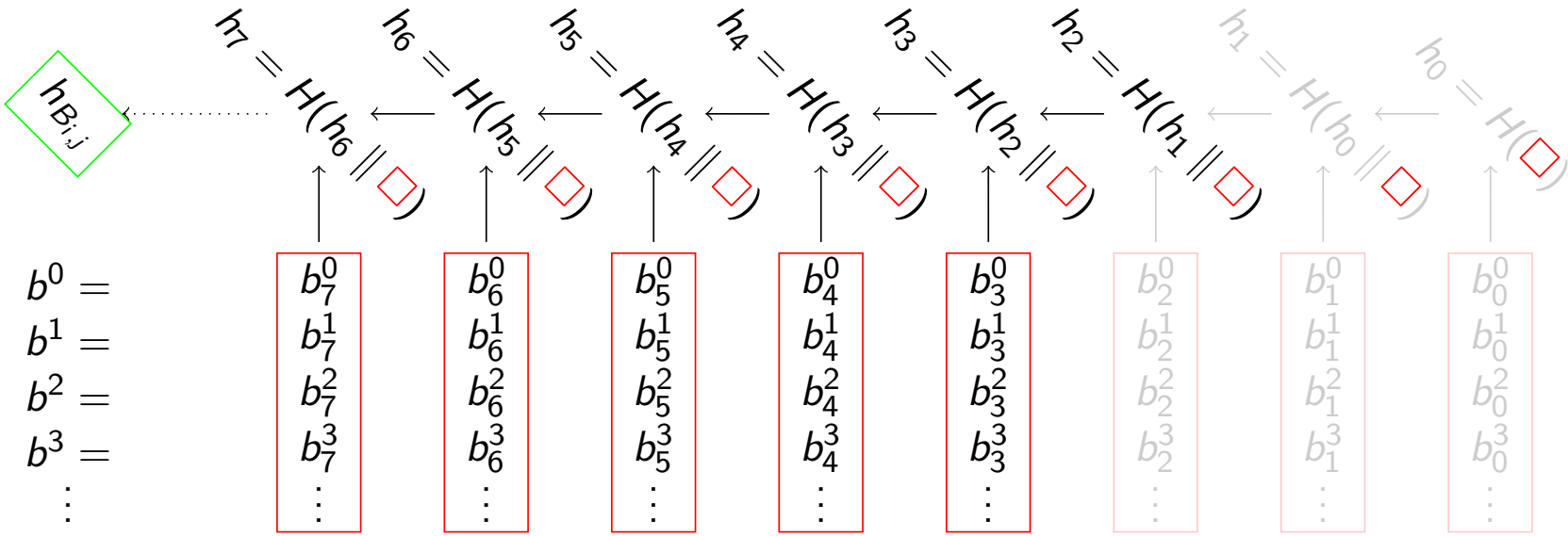
Proof of Security

Or we go one level deeper for each $h_{B_{i,j}}$.



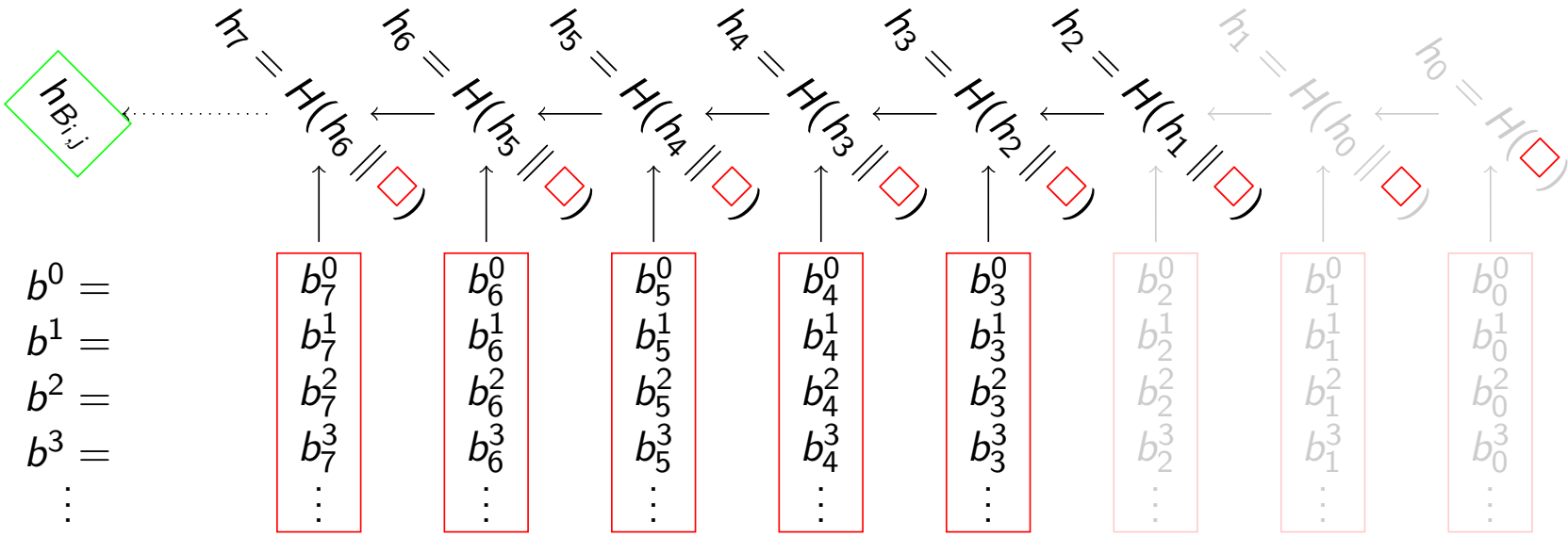
Proof of Security

Or we go one level deeper for each $h_{B_{i,j}}$.



Proof of Security

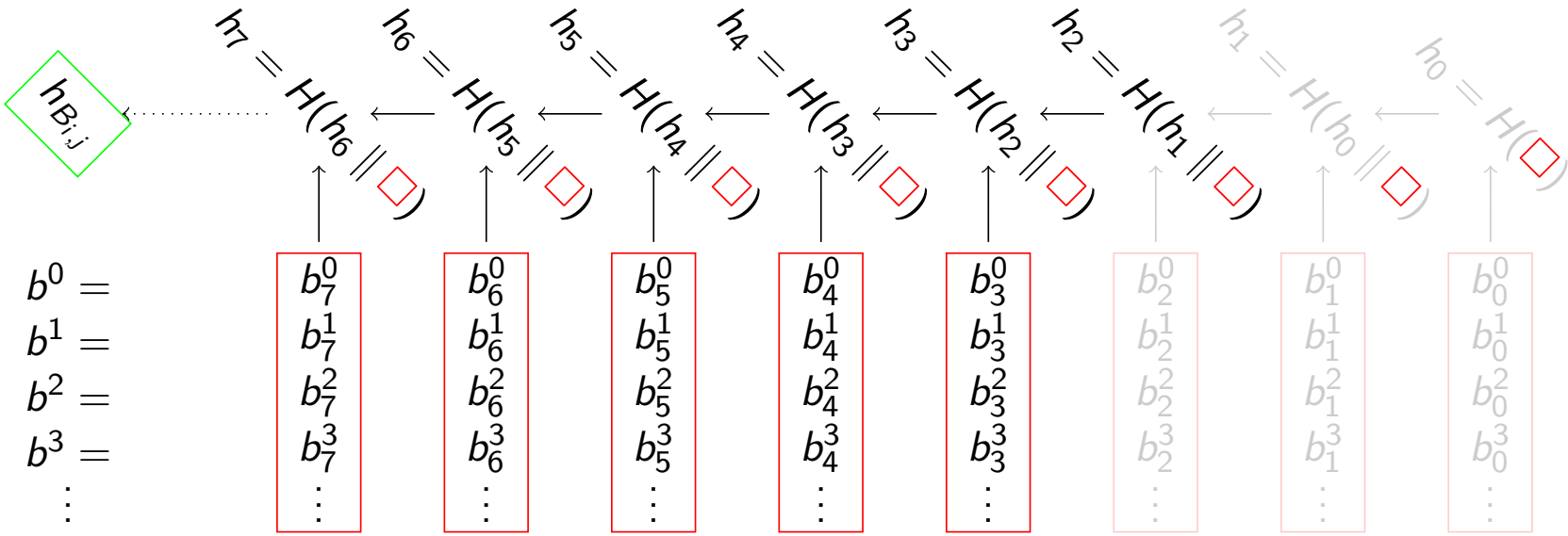
Or we go one level deeper for each $h_{B_{i,j}}$.



Since $i^* \notin \mathrm{CSpan}(i_k, s_k)$ we are guaranteed tho eventually find a difference

Proof of Security

Or we go one level deeper for each $h_{B_{i,j}}$.



Since $i^* \notin \mathrm{CSpan}(i_k, s_k)$ we are guaranteed tho eventually find a difference, and hence a collision for H. $\square$