FSE/tANS Introduction and Implementation

ECSE 6680 Course Project

04/22/2024

Data compression

Facebook/Meta ZSTD - replacing gzip

Apple LZFSE - iPhone, MAC

CRAM - DNA compressor

JPEG XL - replacing JPEG ~ 3× smaller

Saving time, transmission costs,
energy, storage, hardware costs

Data compression

Facebook/Meta ZSTD - replacing gzip

Apple LZFSE - iPhone, MAC

CRAM - DNA compressor

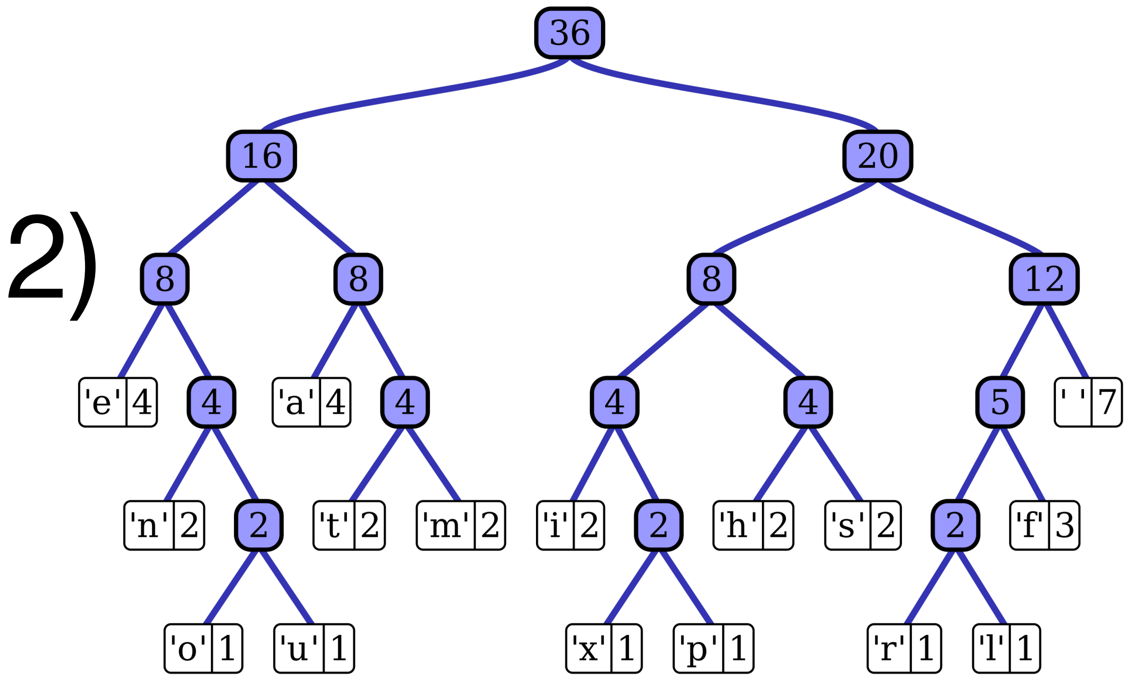
CRAM: The Data Storage Standard For Genomics

JPEG XL - replacing JPEG ~ 3X smaller

Saving time, transmission costs, energy, storage, hardware costs

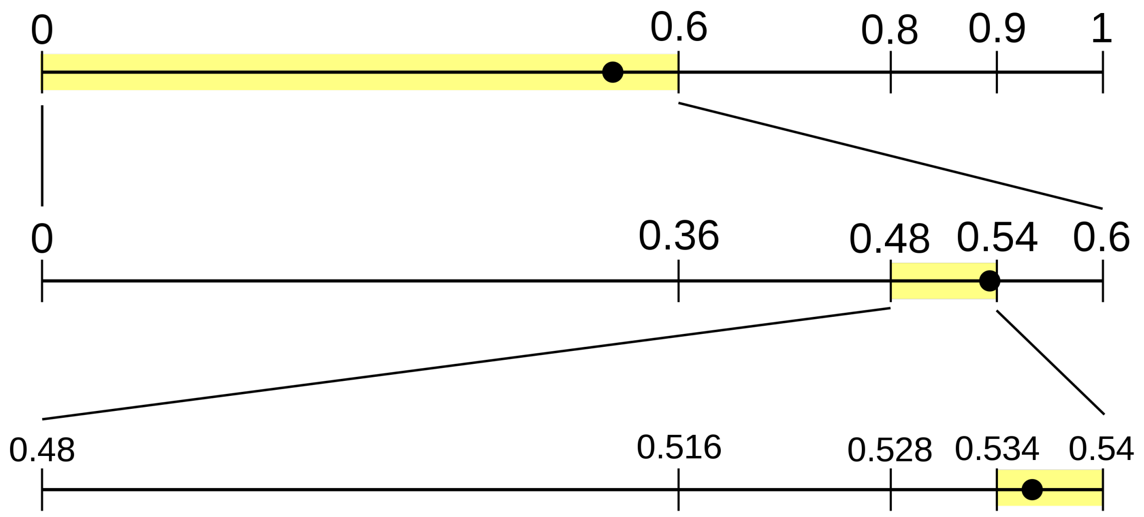
Brief history

Huffman coding (1952)



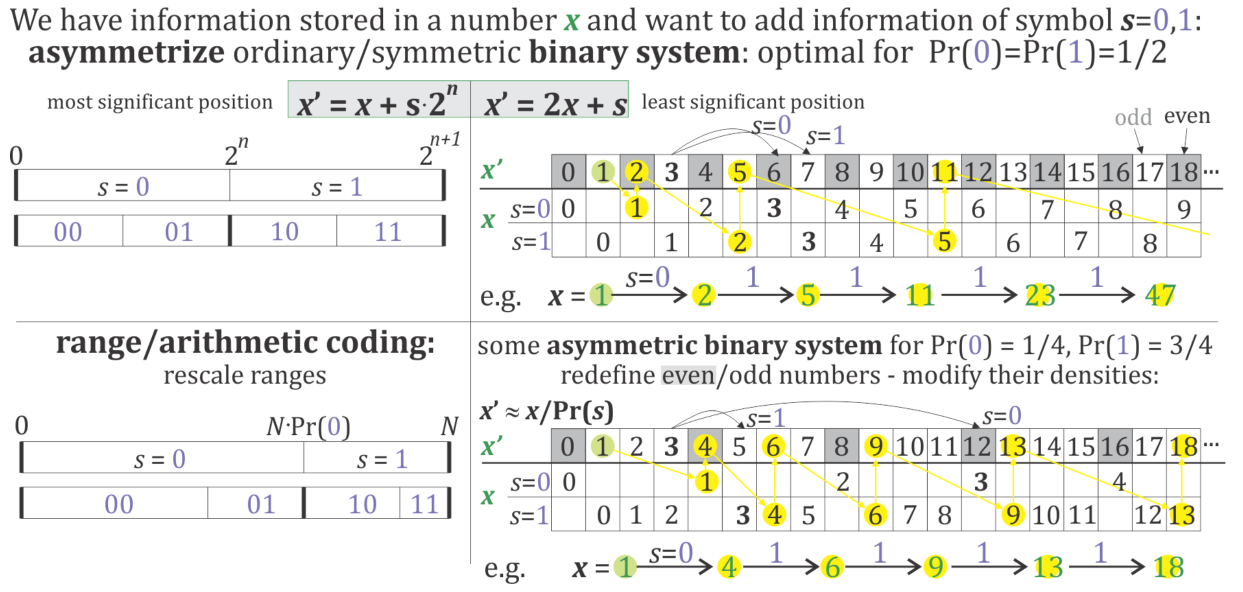
Faster

Arithmetic coding (1976)



Simpler

Asymmetric numeral systems (2006)


We have information stored in a number x and want to add information of symbol s=0,1:
asymmetrize ordinary/symmetric **binary system**: optimal for Pr(0)=Pr(1)=1/2

most significant position $x' = x + s \cdot 2^n$ | $x' = 2x + s$ least significant position

x'		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18…
x	s=0	0		1		2		3		4		5		6		7		8		9
	s=1		0		1		2		3		4		5		6		7		8	

e.g. $x = 1 \xrightarrow{s=0} 2 \xrightarrow{1} 5 \xrightarrow{1} 11 \xrightarrow{1} 23 \xrightarrow{1} 47$

range/arithmetic coding:
rescale ranges

some **asymmetric binary system** for Pr(0) = 1/4, Pr(1) = 3/4
redefine even/odd numbers - modify their densities:

$x' \approx x/\Pr(s)$

x'		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18…
x	s=0	0				1				2				3				4		
	s=1		0	1	2		3	4	5		6	7	8		9	10	11		12	13

e.g. $x = 1 \xrightarrow{s=0} 4 \xrightarrow{1} 6 \xrightarrow{1} 9 \xrightarrow{1} 13 \xrightarrow{1} 18$

Less memory

tANS/FSE

Symmetric numeral system

Let's assume inputs are digits $[0,9]$

We want to encode $[3,2,4,1,5]$ into a natural number

32415 is the answer

Symmetric numeral system

Let’s assume inputs are digits [0,9]

We want to encode [3,2,4,1,5] into a natural number

32415 is the answer

```
# given
data_input = [3,2,4,1,5]
## "encoding" process
x = 0 # ⟵ initial state
x = x*10 + 3 #x = 3
x = x*10 + 2 #x = 32
x = x*10 + 4 #x = 324
x = x*10 + 1 #x = 3241
x = x*10 + 5 #x = 32415 ← final state
```

Symmetric numeral system

Let’s assume inputs are digits $[0,9]$

We want to encode $[3,2,4,1,5]$ into a natural number

32415 is the answer

```
# given
data_input = [3,2,4,1,5]
## "encoding" process
x = 0 # ⟵ initial state
x = x*10 + 3 #x = 3
x = x*10 + 2 #x = 32
x = x*10 + 4 #x = 324
x = x*10 + 1 #x = 3241
x = x*10 + 5 #x = 32415 ← final state
```

If the digits $[0,9]$ have different probabilities than 0.1? $log_2(10) = log_2\frac{1}{0.1} = log_2\frac{1}{p}$

What is ANS?

ANS - store information as a natural number x

How to encode 101111 to natural number 18

What is ANS?

ANS - store information as a natural number x

How to encode 101111 to natural number 18

$\boxed{x' \approx x/\Pr(s)}$ → Coding rule

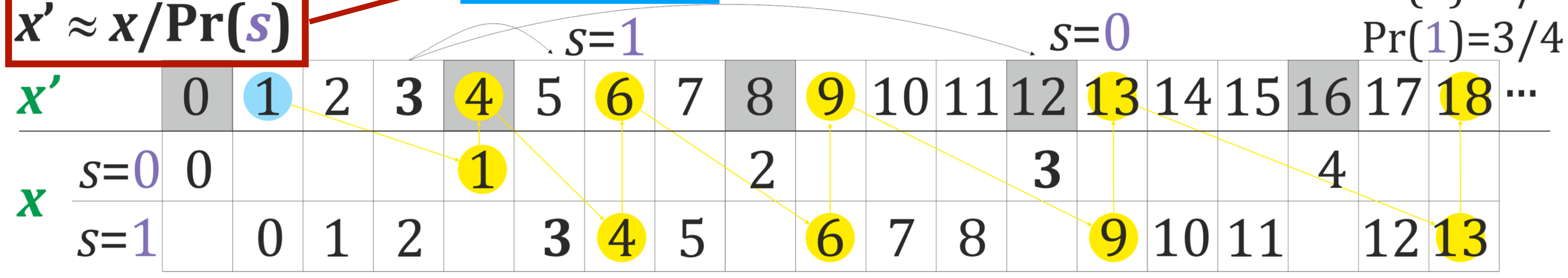
$\Pr(0)=1/4$
$\Pr(1)=3/4$

Probability of next symbol

x': next state

x: previous state

s: symbol that we want to store in x



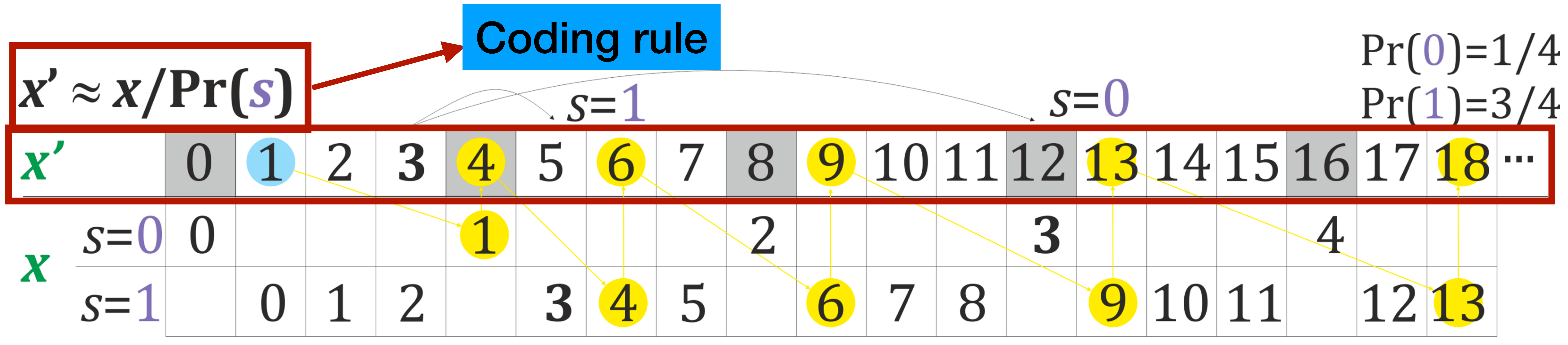
x'		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18 ⋯
x	$s=0$	0				1				2				3				4		
	$s=1$		0	1	2		3	4	5		6	7	8		9	10	11		12	13

e.g. $x = 1 \xrightarrow{s=0} 4 \xrightarrow{1} 6 \xrightarrow{1} 9 \xrightarrow{1} 13 \xrightarrow{1} 18$

What is ANS?

ANS - store information as a natural number x

How to encode 101111 to natural number 18



Coding rule

$x' \approx x/\Pr(s)$

Pr(0)=1/4

Pr(1)=3/4

		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	···
x'		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	···
x	$s=0$	0				1				2				3				4			
	$s=1$		0	1	2		3	4	5		6	7	8		9	10	11		12	13	

e.g. $x = 1 \xrightarrow{s=0} 4 \xrightarrow{1} 6 \xrightarrow{1} 9 \xrightarrow{1} 13 \xrightarrow{1} 18$

Probability of next symbol

x': next state

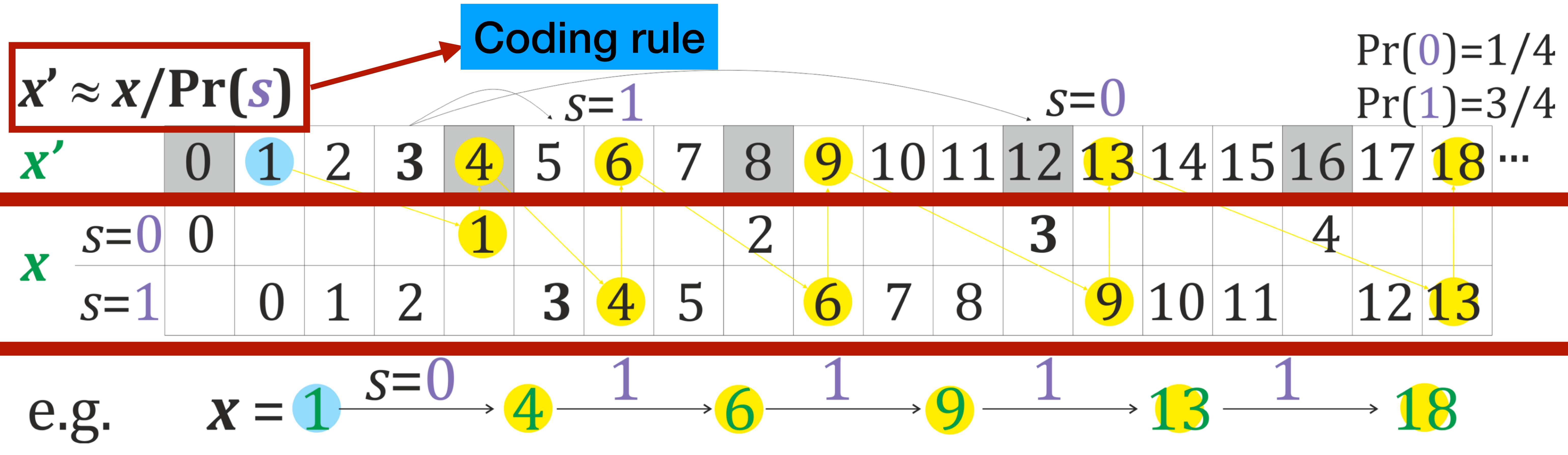
x: previous state

s: symbol that we want to store in x

What is ANS?

ANS - store information as a natural number x

How to encode 101111 to natural number 18



x'		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	···
x	$s=0$	0				1				2				3				4			
	$s=1$		0	1	2		3	4	5		6	7	8		9	10	11		12	13	

e.g. $x = 1 \xrightarrow{s=0} 4 \xrightarrow{1} 6 \xrightarrow{1} 9 \xrightarrow{1} 13 \xrightarrow{1} 18$

Probability of next symbol

x': next state

x: previous state

s: symbol that we want to store in x

What is ANS?

ANS - store information as a natural number x

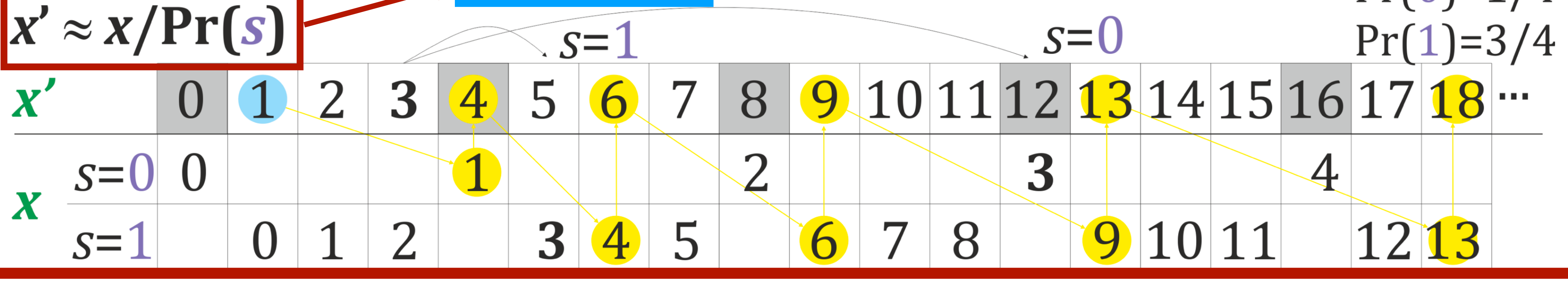
How to encode 101111 to natural number 18

$x' \approx x/\Pr(s)$

Coding rule

Pr(0)=1/4
Pr(1)=3/4

Probability of next symbol



		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	···
x	s=0	0				1				2				3				4			
	s=1		0	1	2		3	4	5		6	7	8		9	10	11		12	13	

x': next state

x: previous state

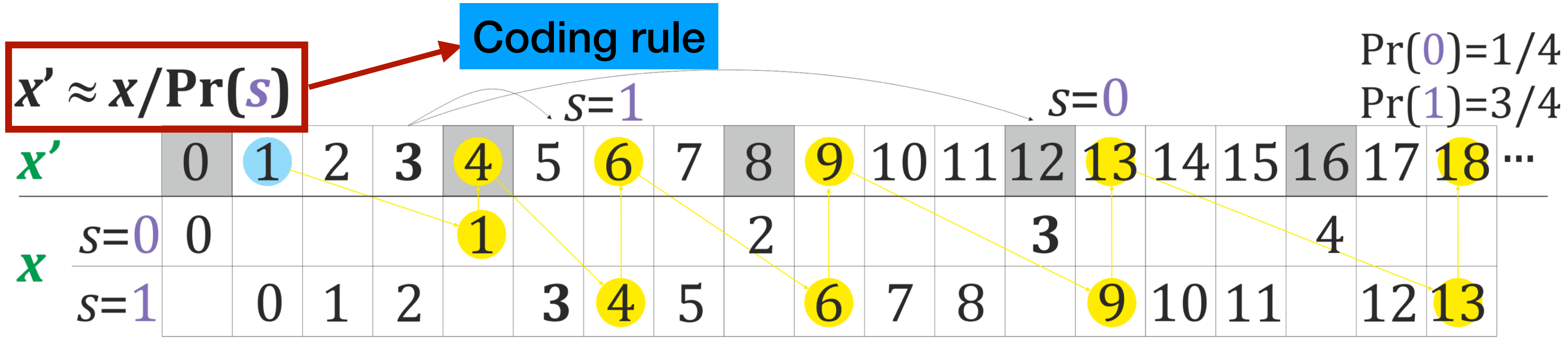
s: symbol that we want to store in x

e.g. $x = 1 \xrightarrow{s=0} 4 \xrightarrow{1} 6 \xrightarrow{1} 9 \xrightarrow{1} 13 \xrightarrow{1} 18$

What is ANS?

ANS - store information as a natural number x

How to encode 101111 to natural number 18



Coding rule

$x' \approx x/\mathrm{Pr}(s)$

s=1

s=0

Pr(0)=1/4

Pr(1)=3/4

		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	⋯
x	s=0	0				1				2				3				4			
	s=1		0	1	2		3	4	5		6	7	8		9	10	11		12	13	

e.g. $x = 1 \xrightarrow{s=0} 4 \xrightarrow{1} 6 \xrightarrow{1} 9 \xrightarrow{1} 13 \xrightarrow{1} 18$

Probability of next symbol

x': next state

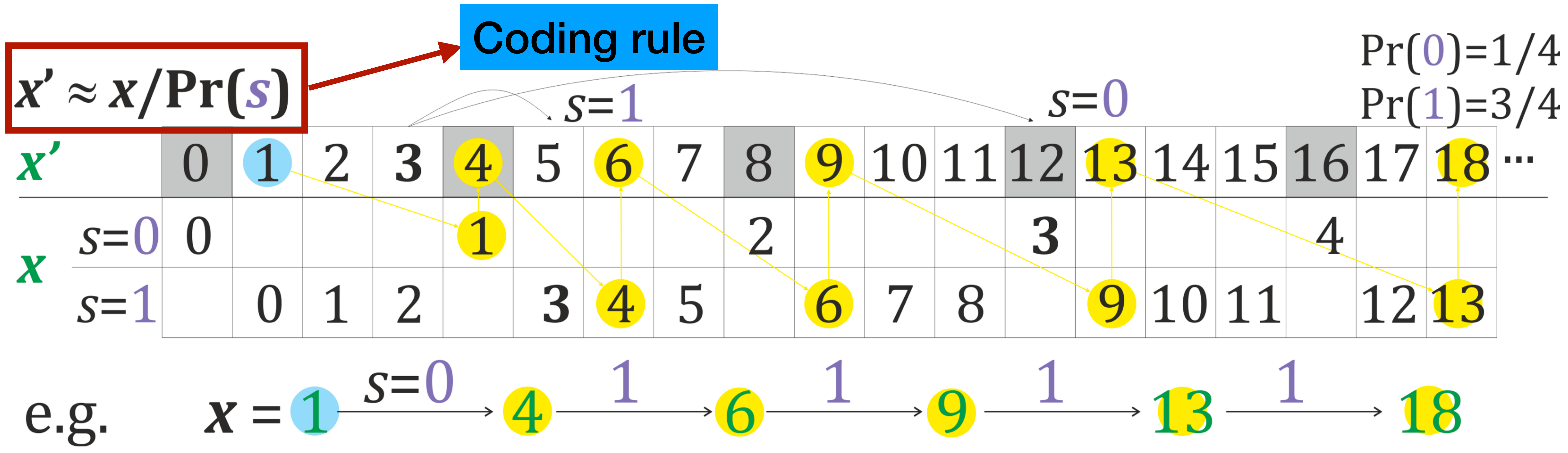
x: previous state

s: symbol that we want to store in x

What is ANS?

ANS - store information as a natural number x

How to encode 101111 to natural number 18



$x' \approx x/\Pr(s)$

Coding rule

$\Pr(0)=1/4$

$\Pr(1)=3/4$

x'	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	···
x $s=0$	0				1				2				3				4			
x $s=1$		0	1	2		3	4	5		6	7	8		9	10	11		12	13	

e.g. $x = 1 \xrightarrow{s=0} 4 \xrightarrow{1} 6 \xrightarrow{1} 9 \xrightarrow{1} 13 \xrightarrow{1} 18$

Probability of next symbol

x': next state

x: previous state

s: symbol that we want to store in x

Grows infinitely $x \to \infty$

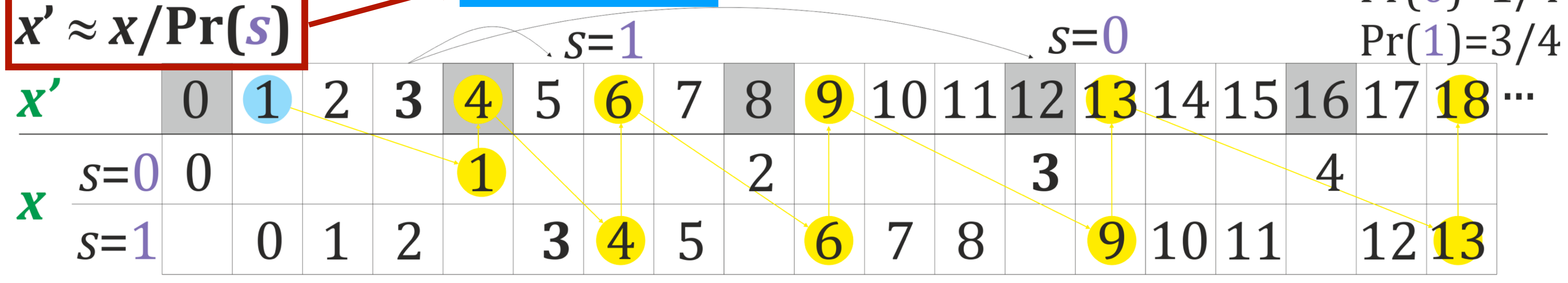
What is ANS?

ANS - store information as a natural number x

How to encode 101111 to natural number 18

$x' \approx x/\Pr(s)$

Coding rule



Pr(0)=1/4
Pr(1)=3/4

		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	…
x'		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	…
x	s=0	0				1				2				3				4			
	s=1		0	1	2		3	4	5		6	7	8		9	10	11		12	13	

e.g. $x = 1 \xrightarrow{s=0} 4 \xrightarrow{1} 6 \xrightarrow{1} 9 \xrightarrow{1} 13 \xrightarrow{1} 18$

```
x=1, s=0, x'=4
x=4, s=1, x'=6
x=6, s=1, x'=9
x=9, s=1, x'=13
…
```

Probability of next symbol

x': next state

x: previous state

s: symbol that we want to store in x

Grows infinitely $x \to \infty$

Fix state range and reuse? - renormalization

To prevent $x \to \infty$, enforce $x \in I = \{L,2L-1\}, L = 2^n$

Fix state range and reuse? - renormalization

To prevent $x \to \infty$, enforce $x \in I = \{L,2L-1\}, L = 2^n$

By streaming: transmitting lowest bits to bitstream

Fix state range and reuse? - renormalization

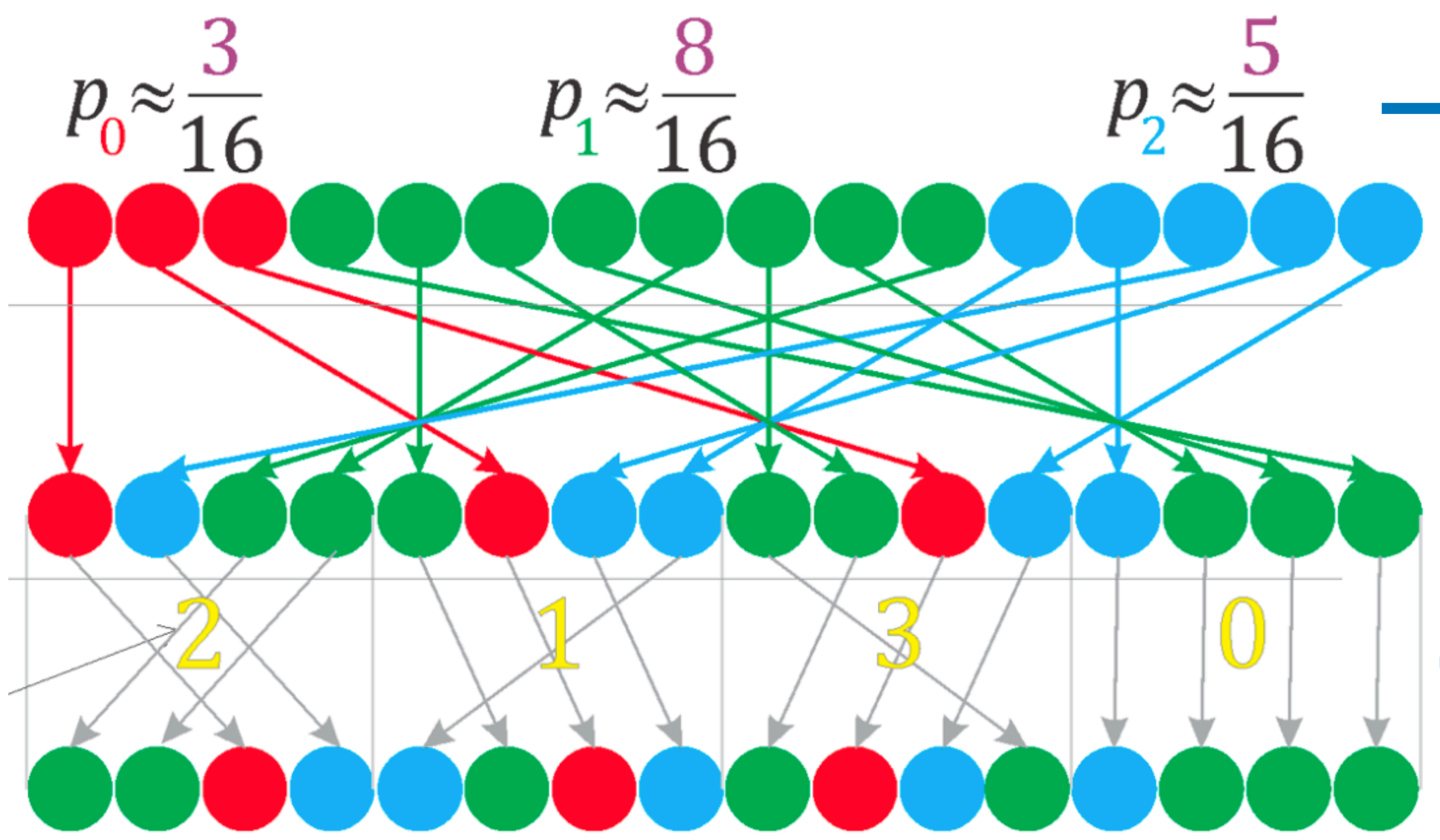
To prevent $x \to \infty$, enforce $x \in I = \{L, 2L - 1\}, L = 2^n$

By streaming: transmitting lowest bits to bitstream

```
# input state
x_prev = 22 = 10110b
# stream out bits from x_prev until we are sure base_step(x_shrunk,s) in [8,15]
x_shrunk = 10110b = 22, out_bits = ""
x_shrunk = 1011b = 11, out_bits = "0"
```

tabled ANS/Finite state entropy

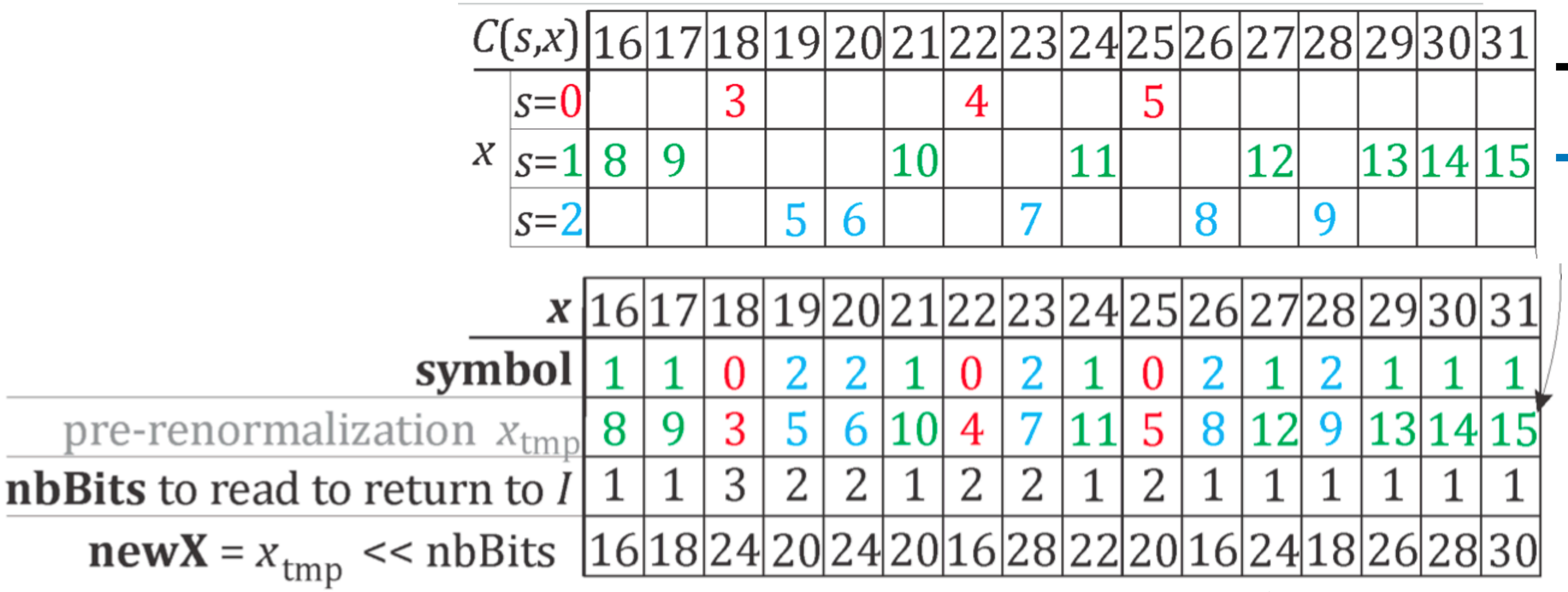
Another Example: encode 0, 1, 2 with probabilities



$p_0 \approx \frac{3}{16}$
$p_1 \approx \frac{8}{16}$
$p_2 \approx \frac{5}{16}$
2
1
3
0

$L = 16$

Shuffle for secure



	$C(s,x)$	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
x	$s=0$			3				4			5						
	$s=1$	8	9				10			11			12		13	14	15
	$s=2$				5	6			7			8		9			

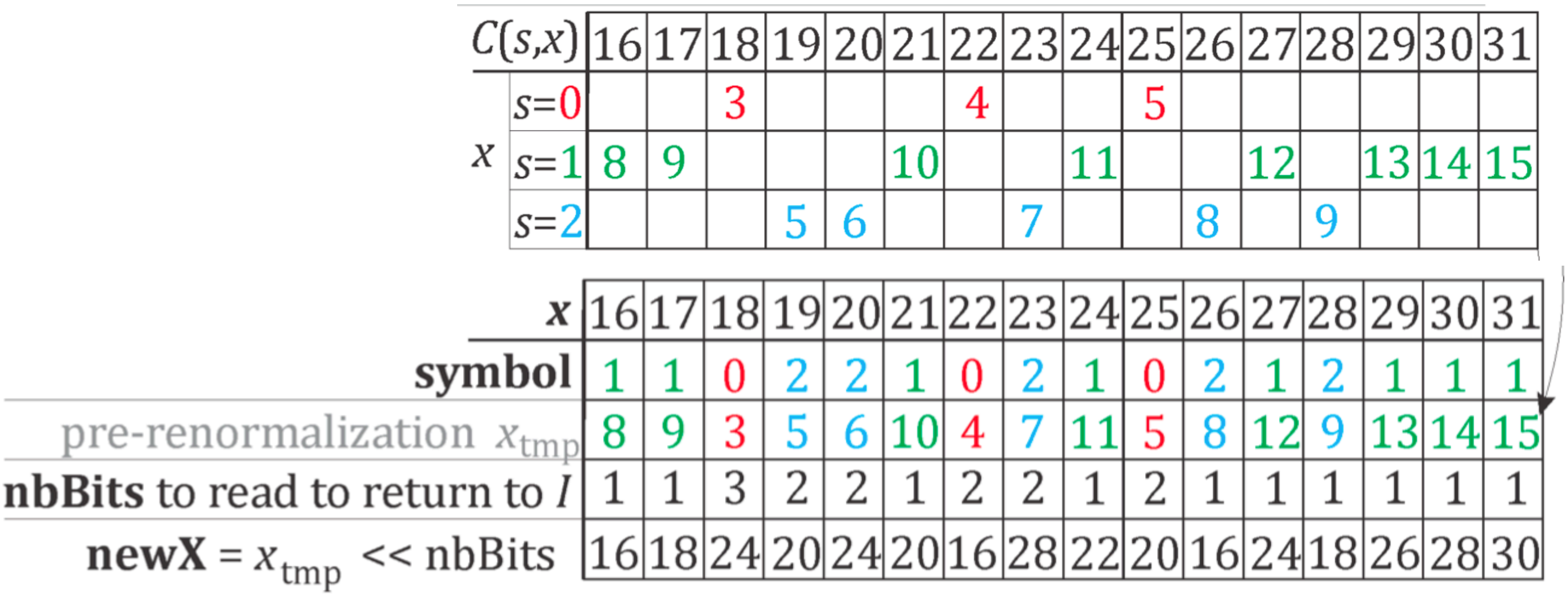
table

Generate the symbol-state mapping

x	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
symbol	1	1	0	2	2	1	0	2	1	0	2	1	2	1	1	1
pre-renormalization x_{tmp}	8	9	3	5	6	10	4	7	11	5	8	12	9	13	14	15
nbBits to read to return to I	1	1	3	2	2	1	2	2	1	2	1	1	1	1	1	1
newX = x_{tmp} << nbBits	16	18	24	20	24	20	16	28	22	20	16	24	18	26	28	30

tabled ANS/Finite state entropy-cont.

Another Example: encode 0, 1, 2 with probabilities



$C(s,x)$		16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
	$s=0$			3				4			5						
x	$s=1$	8	9				10			11			12		13	14	15
	$s=2$				5	6			7			8		9			

x	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
symbol	1	1	0	2	2	1	0	2	1	0	2	1	2	1	1	1
pre-renormalization x_{tmp}	8	9	3	5	6	10	4	7	11	5	8	12	9	13	14	15
nbBits to read to return to I	1	1	3	2	2	1	2	2	1	2	1	1	1	1	1	1
newX = x_{tmp} << nbBits	16	18	24	20	24	20	16	28	22	20	16	24	18	26	28	30

Hardware implementation

1. Arithmetic Package (arithmetic_pkg.vhd)

Provides general arithmetic functions that are essential across various modules, particularly those that compute logarithmic and ceiling functions.

2. Entropy Package (entropy_pkg.vhd)

Manages the entropy coding part of the system, offering functions to convert literal lengths, match lengths, and offsets into codes that are used for efficient storage and retrieval.

3. Compressor Package (compressor_pkg.vhd)

Contains utilities for data compression, including hash functions and mechanisms to manage data streams and buffers effectively.

4. FSE Tables Package (fse_tables_pkg.vhd)

Stores all necessary tables for the FSE encoding process, such as state transition tables and symbol conversion tables, which are crucial for the encoder's state management.

5. Bitstream Merger (bitstream_2to1_merger.vhd)

A utility that merges two bitstreams into a single output stream. This functionality is crucial for combining multiple streams of encoded data into a unified format.

6. FSE to Single Bitstream Converter (fse2single_bitstream.vhd)

Converts the multi-part FSE encoded data into a single continuous bitstream, simplifying the output for transmission or storage.

7. FSE Encoder (fse_encoder.vhd)

The core module that interfaces with the entropy, arithmetic, and FSE tables packages to perform the actual data encoding. It processes input sequences and produces encoded bitstreams, along with metadata about the encoding state.