Technion

Electronics
Computers
Communications

Information in Storage Devices

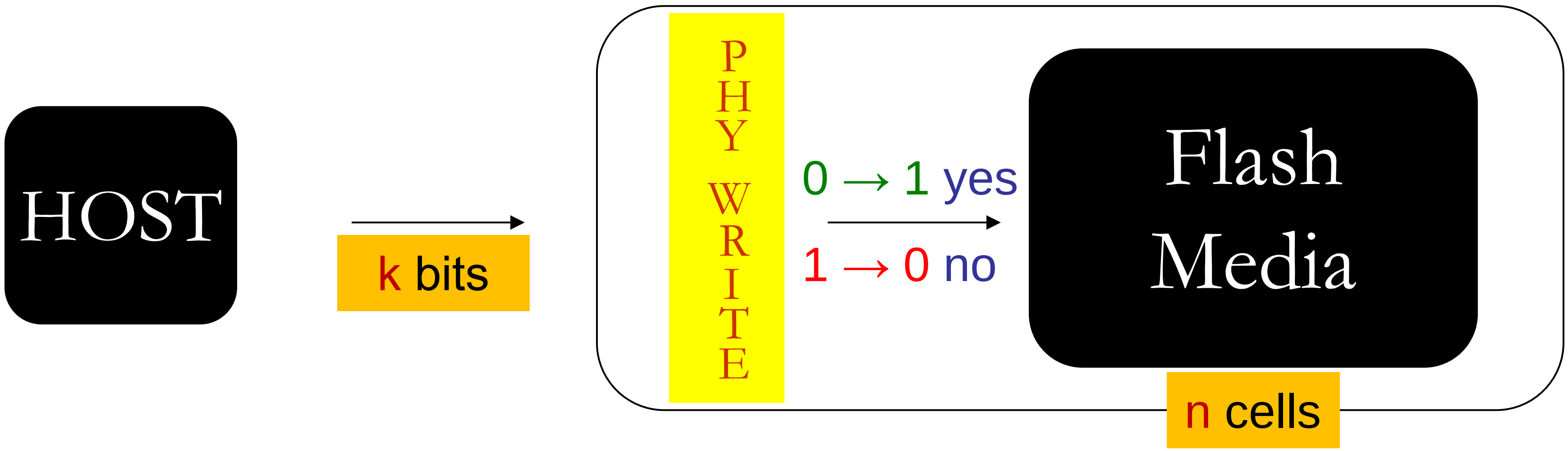
049063 – EE Department, Technion

LECTURE 8: MULTI-LEVEL RE-WRITE CODES

Re-Write Codes

(n,k,t) WOM code:

Write k bits t times, on n cells



$$\underbrace{W\ W\ \dots\ W}_{t\ \text{writes}}\ E\ W\ W\ \dots\ W\ E\ \dots$$

Multi-Write (=Re-Write) Access

- Example t = 2
  - Writes: 0,1,2,4,5,6,8,9,10,12,0,4,8,12

~~0~~	~~4~~	~~8~~	~~12~~	0
1	5	9	13	4
2	6	10	14	8
3	7	11	15	12

~~4~~	Invalid
5	Written twice

Re-Write Code Specification

1) Decoding Function

$$\psi\left(c_1,\cdots,c_n\right)=\left(y_1,\cdots,y_k\right)$$

$$c_i \in \{0,1\}$$

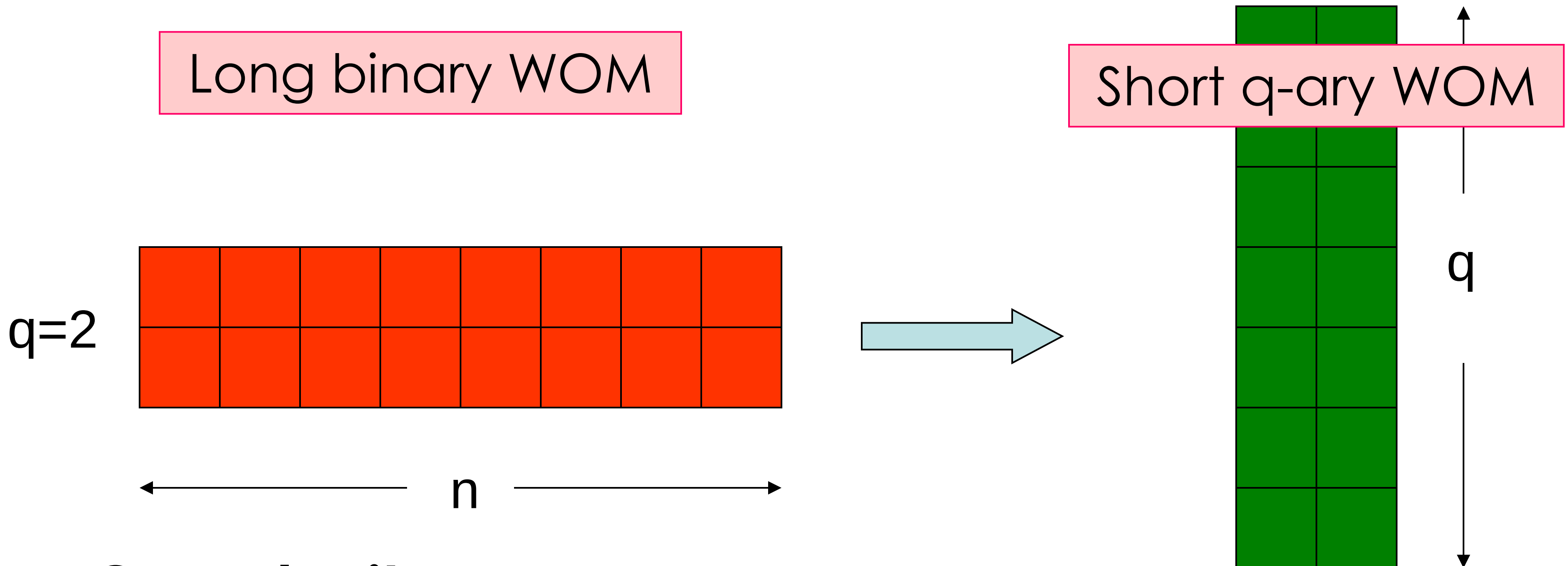
2) Update Function

1 ~~→~~ 0

$$\mu\left(\underbrace{c_1,\cdots,c_n}_{\text{current cell values}};\underbrace{y_1,\cdots,y_k}_{\text{new data}}\right)=\left(\underbrace{c'_1,\cdots,c'_n}_{\text{new cell values}}\right)$$

Binary → q-ary

- **Storage rate:** q-ary WOM codes need lower redundancy for the same number of writes



small n

- **Complexity:**

Exponential in n

Polynomial in q

Multi-Level Re-Write Codes

1) Decoding Function

$$\psi\left(c_1,\cdots,c_n\right)=\left(y_1,\cdots,y_k\right)$$

$$c_i\in\{0,\dots,q-1\}$$

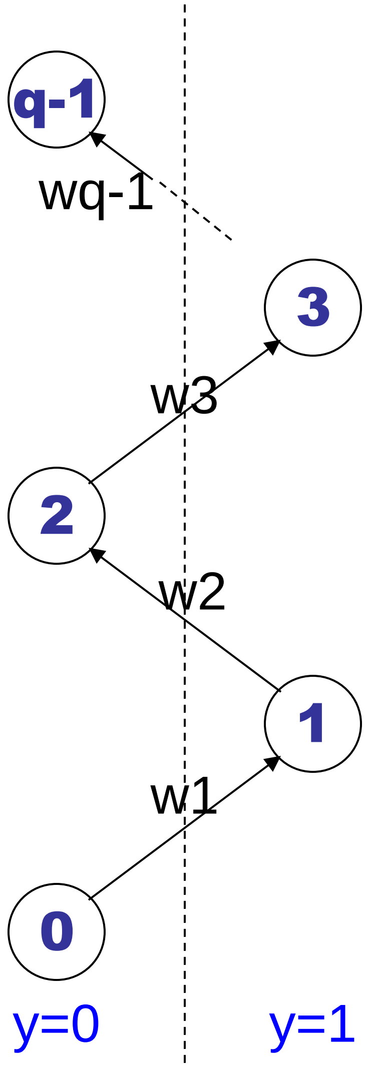
2) Update Function

$i \not\rightarrow <i$

$$\mu\left(\underbrace{c_1,\cdots,c_n}_{\text{current cell values}};\ \underbrace{y_1,\cdots,y_k}_{\text{new data}}\right)=\left(\underbrace{c'_1,\cdots,c'_n}_{\text{new cell values}}\right)$$

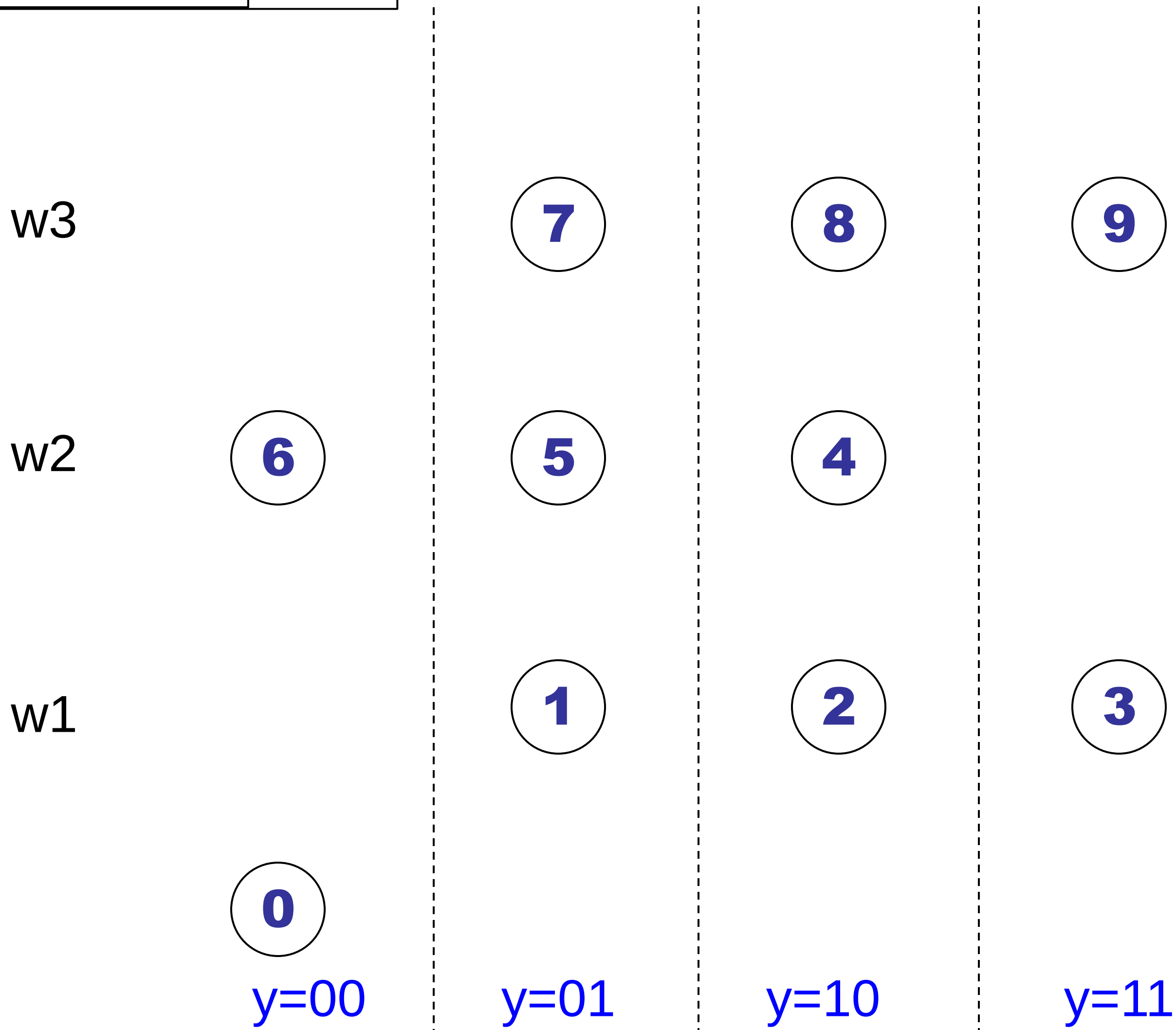
Single-Cell q-ary Code

n=1, k=1, t=q-1



Single-Cell q-ary Code

n=1, k=2, t=? (q-1)/3



Single-Cell, General k

n=1, k, $t=(q-1)/(2^k-1)$

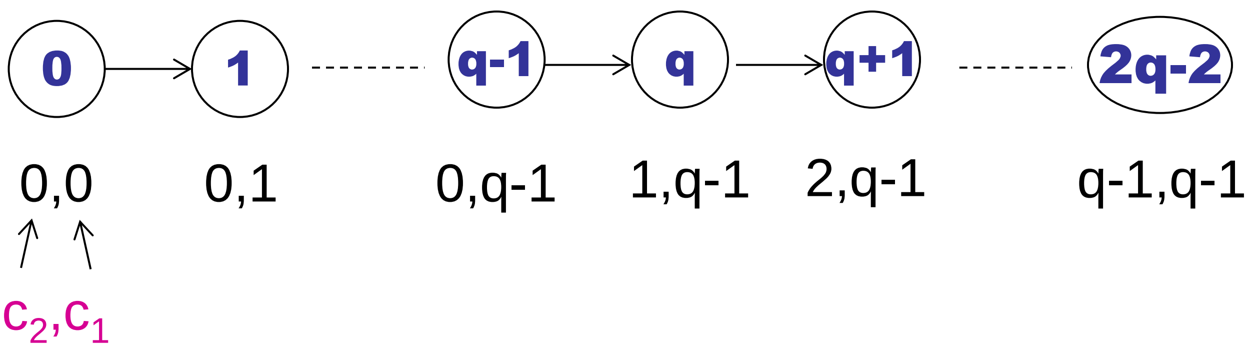
t is inverse exponential in k

2 Cells, k=3

n=2, k=3

Option 1: concatenate

2 x q-ary cells → 1 x 2q-1 ary cell



$$t = \left\lfloor \frac{2q-2}{2^3-1} \right\rfloor = \left\lfloor \frac{2}{7}(q-1) \right\rfloor$$

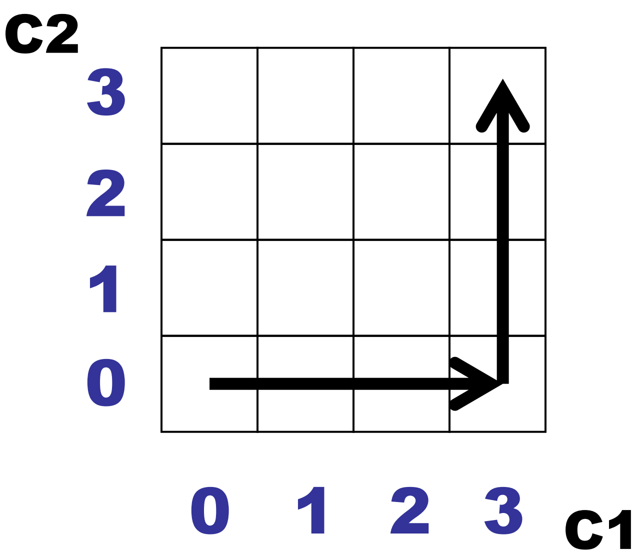
2 Cells, k=3

n=2, k=3

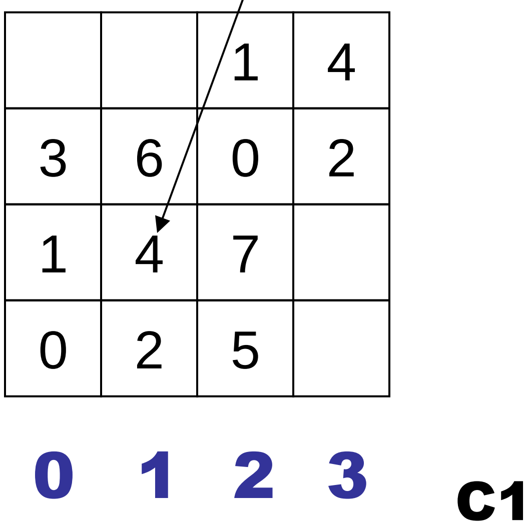
Option 2: 2D code

k=3

$$\{0,1,\cdots,7\} \leftrightarrow (y_1, y_2, y_3)$$



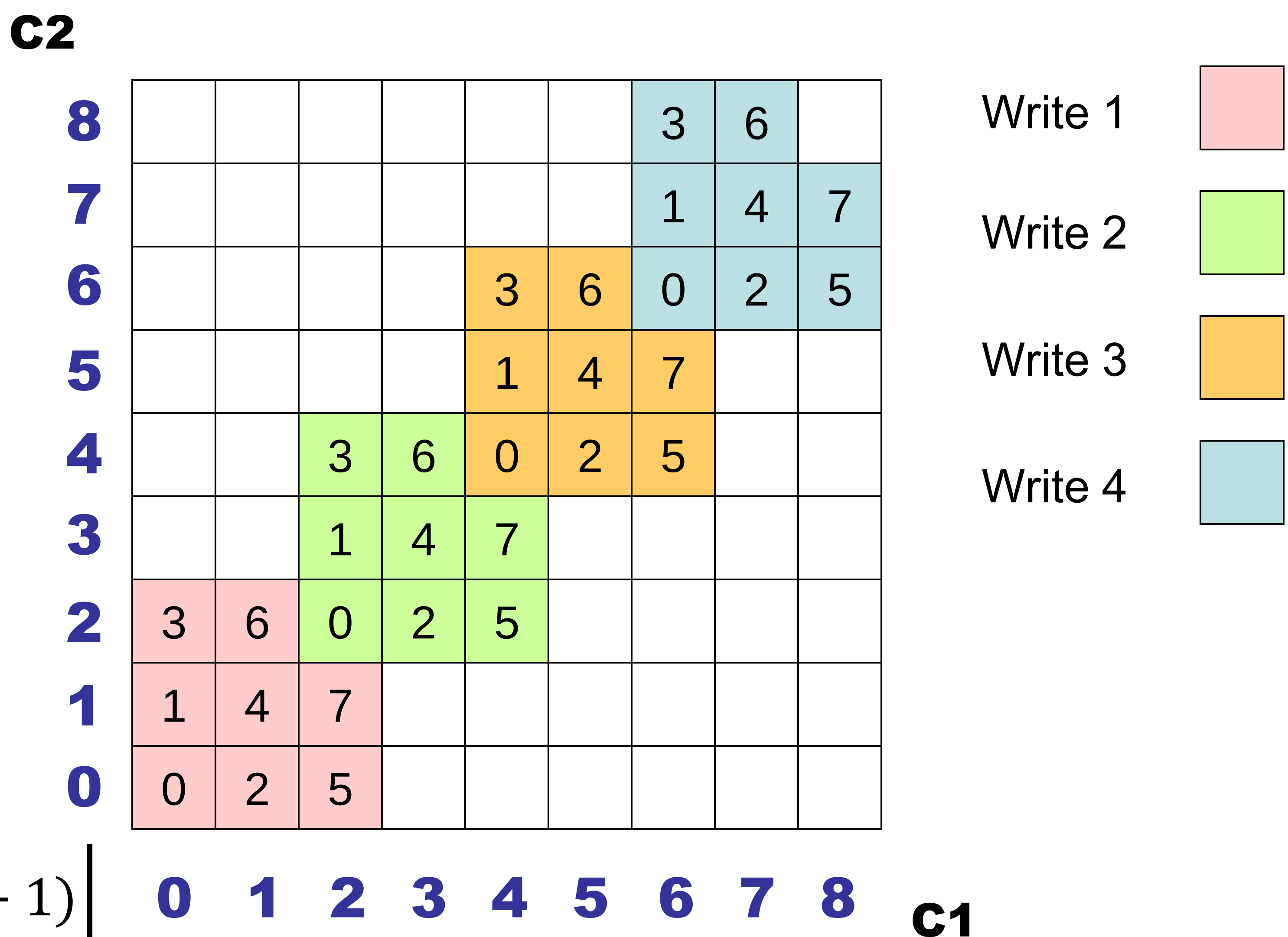
Option 1



Option 2

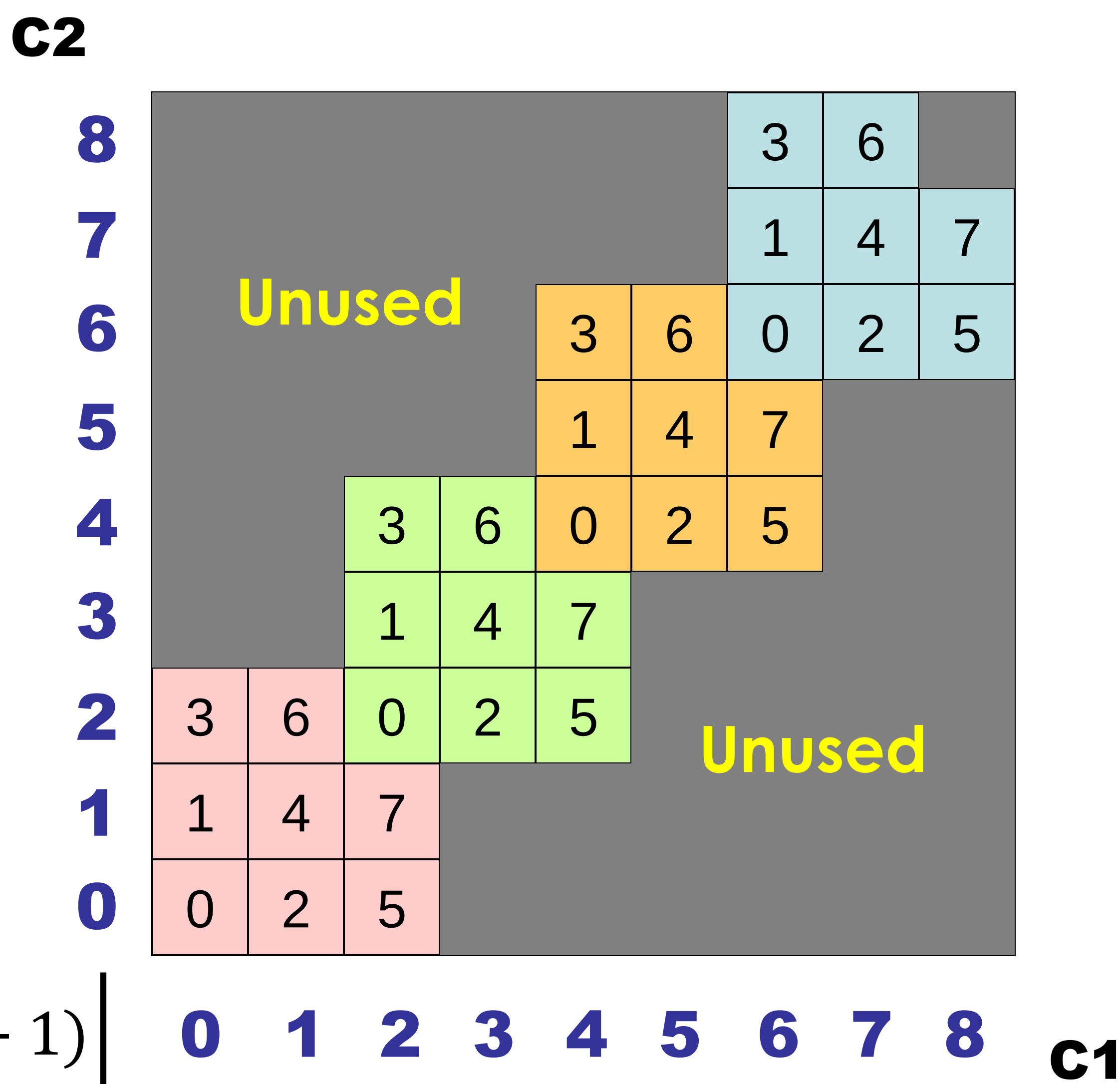
$t = ?$

2 Cells, k=3: 2D Code



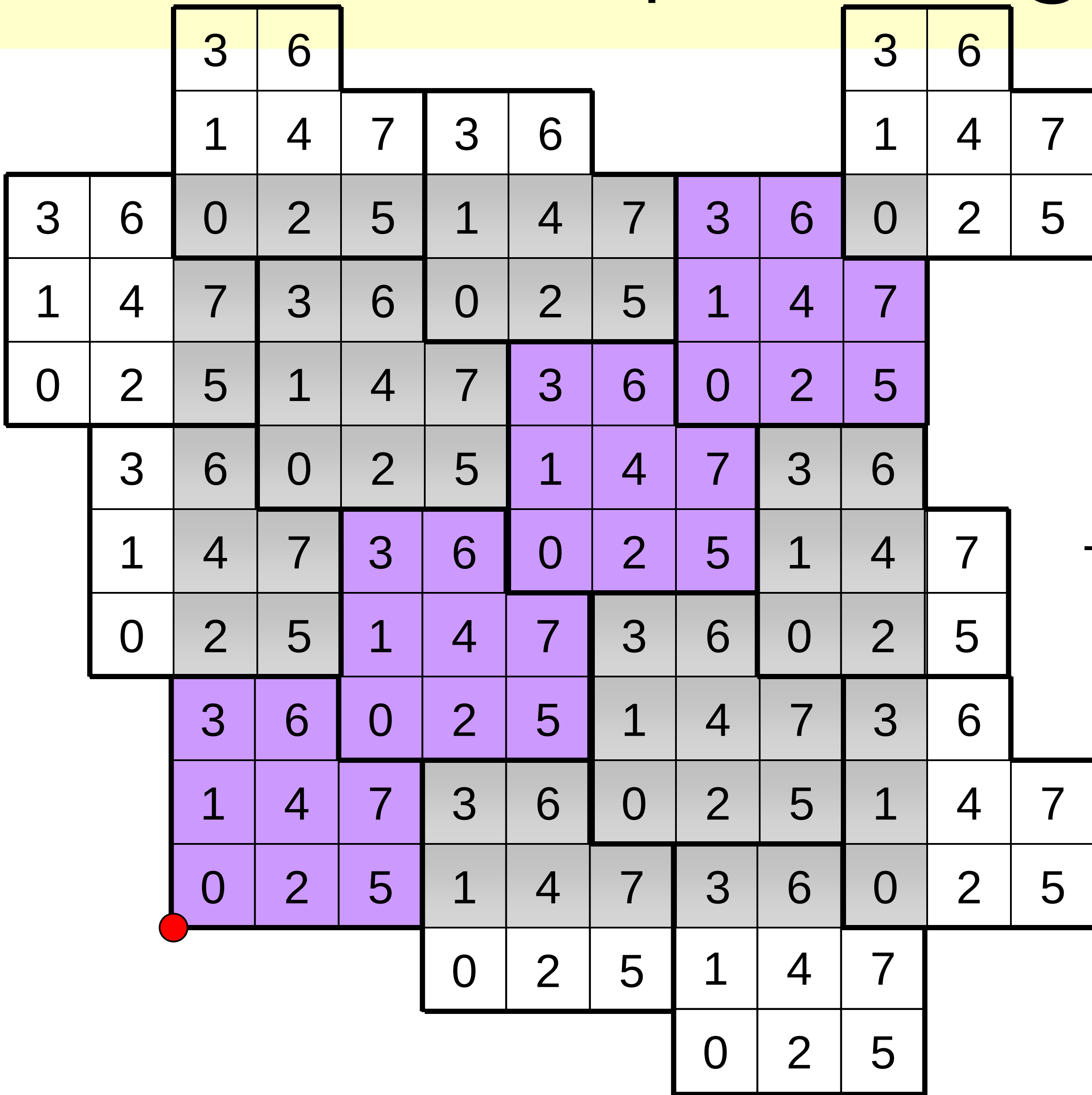
$$t = \left\lfloor \frac{1}{2}(q-1) \right\rfloor$$

Can We Do Better?



$$t = \left\lfloor \frac{1}{2}(q-1) \right\rfloor$$

Step 1: Tiling



<u>Lattice Tiling:</u>

$$v_1 = (2,2)$$

$$v_2 = (3,-1)$$

Translate ● to:

$$a_1 v_1 + a_2 v_2$$

a_1, a_2 integers

t=4 Writes with q=8 Levels

c2									
~~8~~	0	2	5	1	4	7	3	6	0
7	7	3	6	0	2	5	1	4	7
6	5	1	4	7	3	6	0	2	5
5	6	0	2	5	1	4	7	3	6
4	4	7	3	6	0	2	5	1	4
3	2	5	1	4	7	3	6	0	2
2	3	6	0	2	5	1	4	7	3
1	1	4	7	3	6	0	2	5	1
0	0	2	5	1	4	7	3	6	0
	0	1	2	3	4	5	6	7	~~8~~

c1

t=4 Writes with q=8 Levels

c2 \ c1	0	1	2	3	4	5	6	7	
7	7	3	6	0	2	5	1	4	
6	5	1	4	7	3	6	0	2	
5	6	0	2	5	1	4	7	3	
4	4	7	3	6	0	2	5	1	4
3	2	5	1	4	7	3	6	0	
2	3	6	0	2	5	1	4	7	
1	1	4	7	3	6	0	2	5	
0	0	2	5	1	4	7	3	6	

Problem:

2-increment x4

Example, write

→ 7 → 5 → 6 → 4

t=4 Writes with q=8 Levels

c2								
7	7	3	6	0	2	5	1	4
6	5	1	4	7	3	6	0	2
5	6	0	2	5	1	4	7	3
4	4	7	3	6	0	2	5	1
3	2	5	1	4	7	3	6	0
2	3	6	0	2	5	1	4	7
1	1	4	7	3	6	0	2	5
0	0	2	5	1	4	7	3	6
	0	1	2	3	4	5	6	7

c1

Solution:

~~Nearest~~ → Balanced

→ 7 → 5 → 6 → 4

Lattice Equivalence → Balance

7 → 5 → 6 → 4

$\sum$ (2,1) (2,1) (2,1) (2,1) = (8,3)  Nearest

$\sum$ (2,1) (2,1) (2,1) (1,4) = (7,7)  Balanced

Lattice equivalent

$(1,4) = (2,1) + (-1,3) =$

$(2,1) + v_1 - v_2$

2D with Tiling

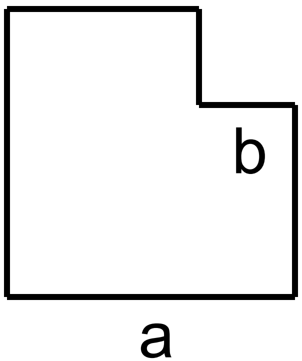
- k=3: $t = \left\lfloor \frac{4}{7}(q-1) \right\rfloor$

- General odd k, t=4:

$$q = \underline{6} \cdot 2^{\frac{k-1}{2}} - 3 \quad \longrightarrow \quad q = \underline{5.5} \cdot 2^{\frac{k-1}{2}} - 3$$

No tiling

Tiling



q-ary WOM Capacity

Theorem: for each q-ary cell, the total information rate in t writes satisfies

$$R_{sum} \le C(q,t) = log_2 \binom{q+t-1}{t} \quad \text{[bits]}$$

Proof idea:
How many ways to divide q-1 (increments) among t+1 sets with sizes >=0 ?

Upper bound on the number of writes:

$$tk \le n \, log_2 \binom{q+t-1}{t}$$

Optimal fixed-rate codes?

- <u>Problem:</u> capacity upper bound likely not tight
  - Allows variable-rate codes
  - Gap is proven for binary codes

Fixed-Rate Upper Bound

Theorem (n=2):
Let s be an integer. If $2^k > s(s+1)/2$, then the number of writes satisfies

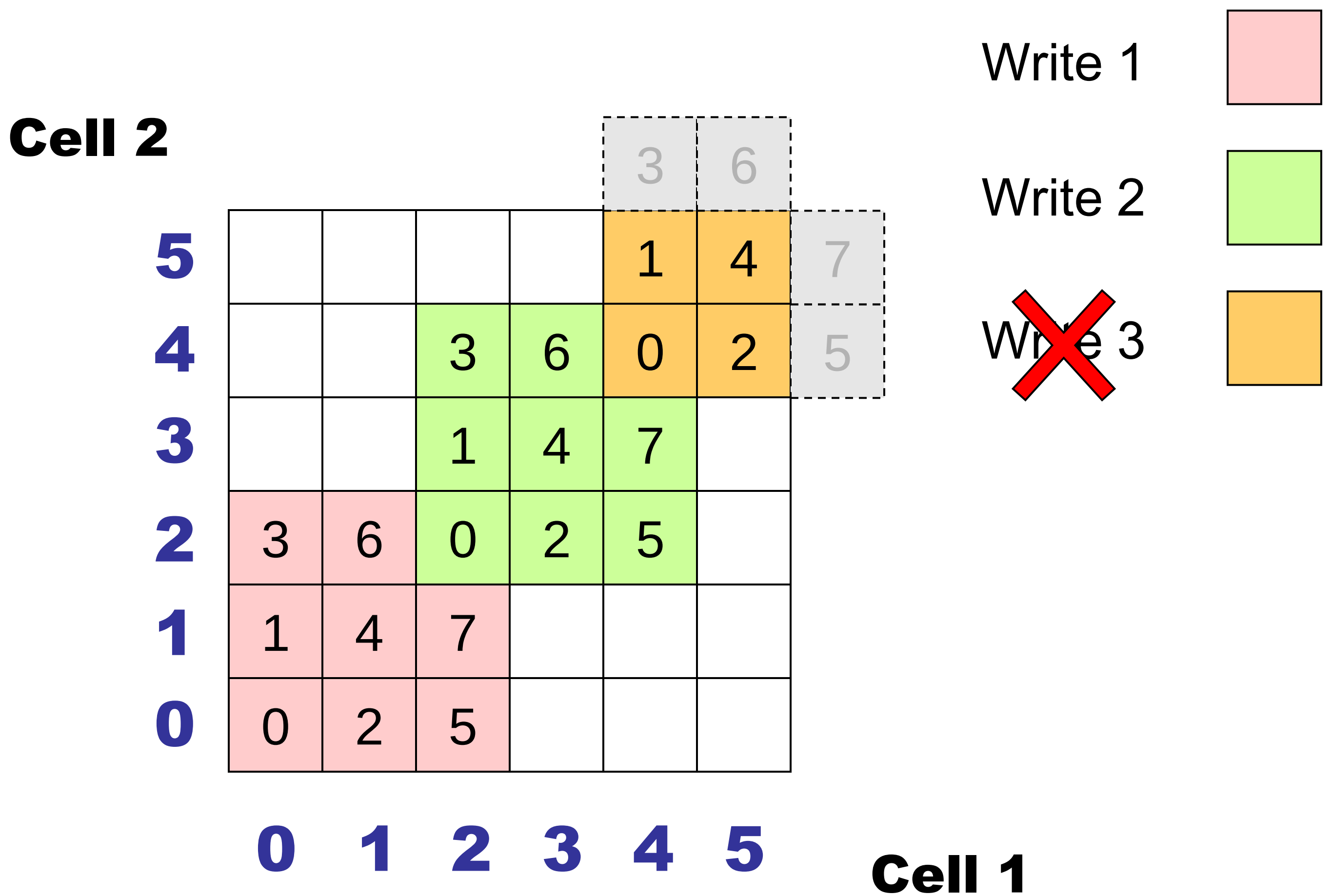
$$t \le \left\lfloor \frac{2(q-1)}{s} \right\rfloor$$

Proof idea:
With k input bits, there exists a worst-case input that increments the sum of levels by at least s.

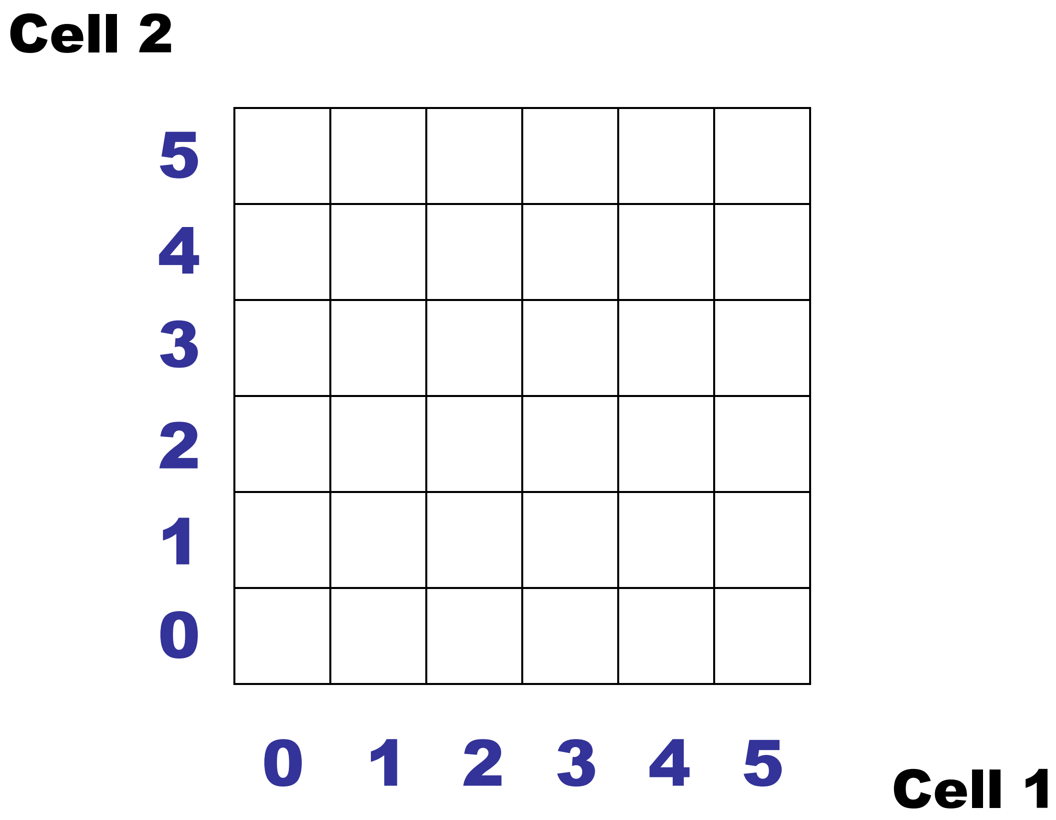
For k=3:
Any code satisfies

$$t \le \left\lfloor \frac{2(q-1)}{3} \right\rfloor$$

Can We Do n=2, k=3, t=3 with q=6?



Can we write 3 times with q=6?



Step 1: easy

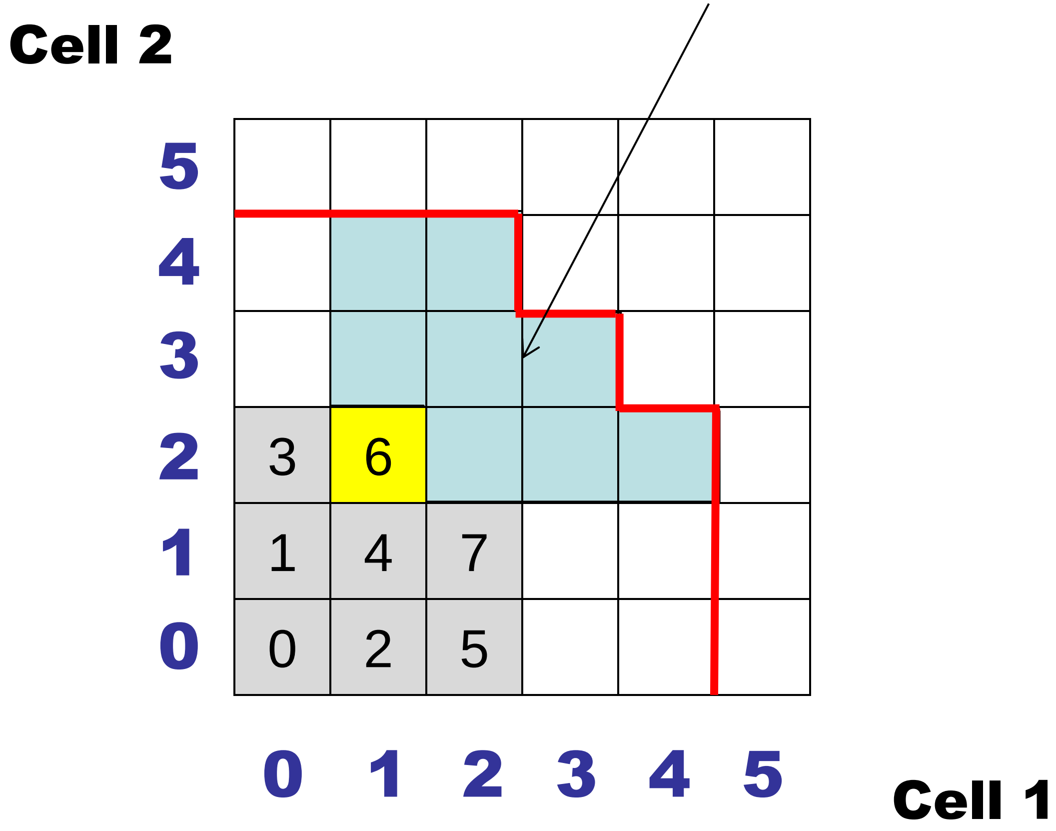
Cell 2

5						
4						
3						
2	3	6				
1	1	4	7			
0	0	2	5			
	0	**1**	**2**	**3**	**4**	**5**

Cell 1

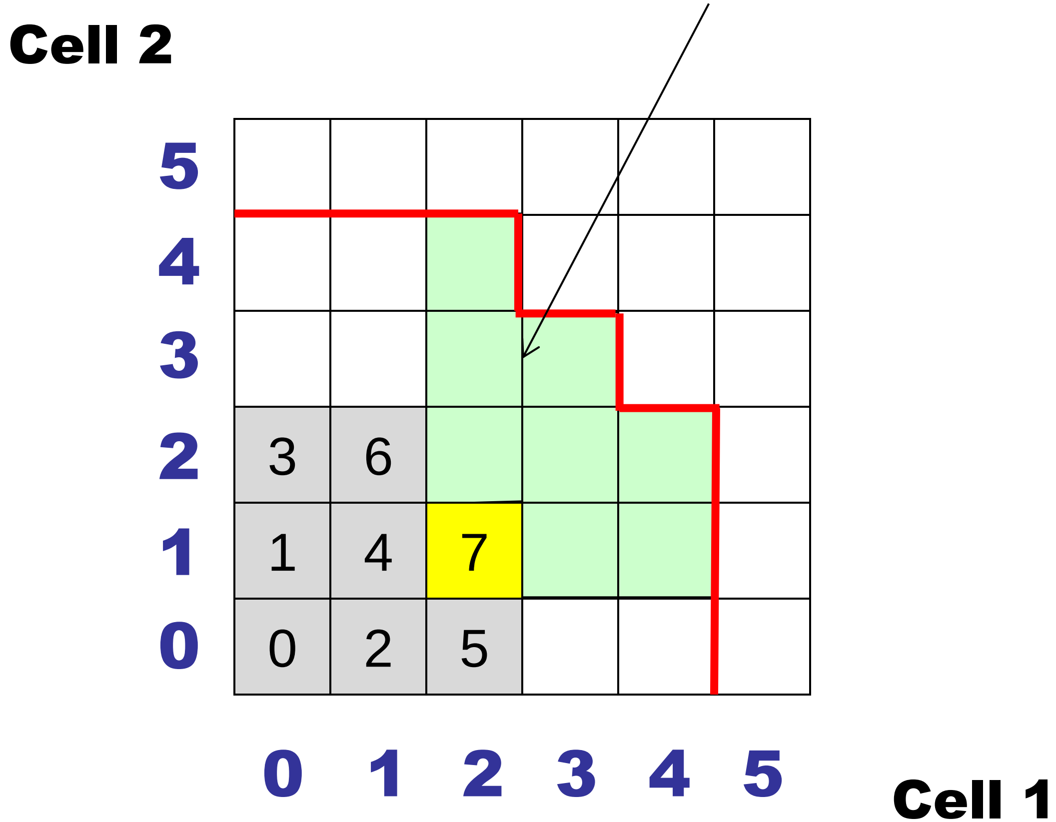
Step 2: 2nd write Sudoku

All numbers 0,1,2,3,4,5,7 in 8 positions

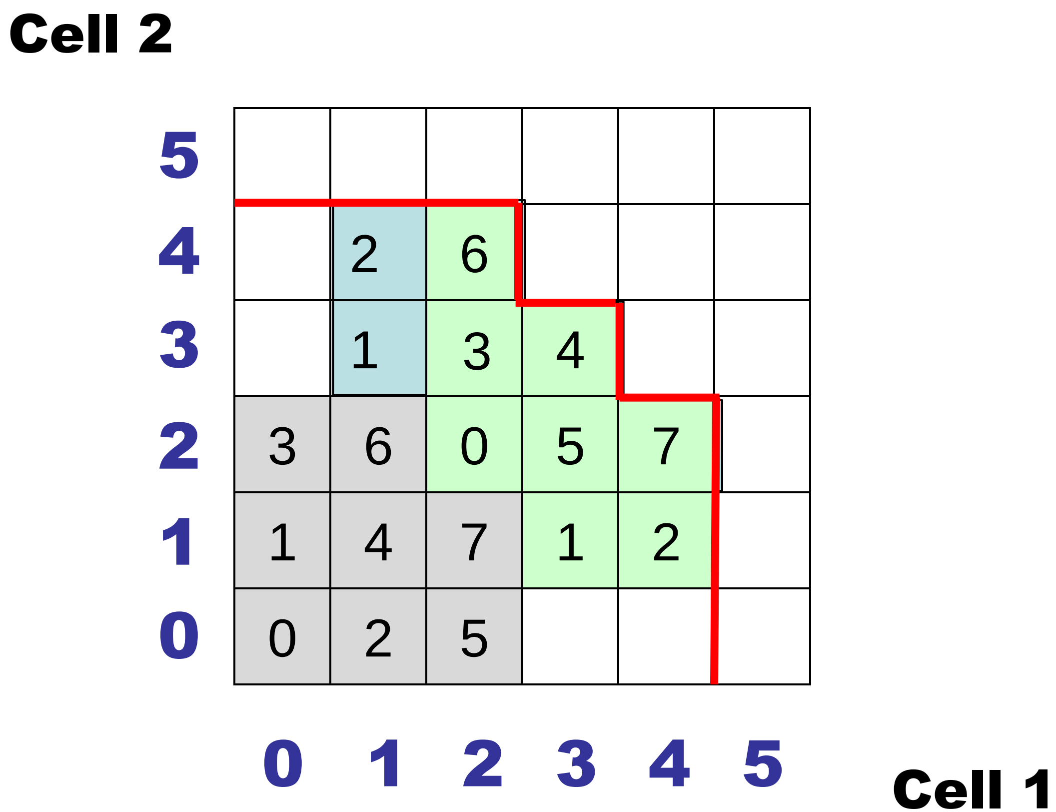


Step 2: 2nd write Sudoku

All numbers 0,1,2,3,4,5,6 in 8 positions



2nd write Solution



Step 3: 3rd write Sudoku

Cell 2

5						
4		2	6			
3		1	3	4		
2	3	6	0	5	7	
1	1	4	7	1	2	
0	0	2	5			
	0	1	2	3	4	5

Cell 1

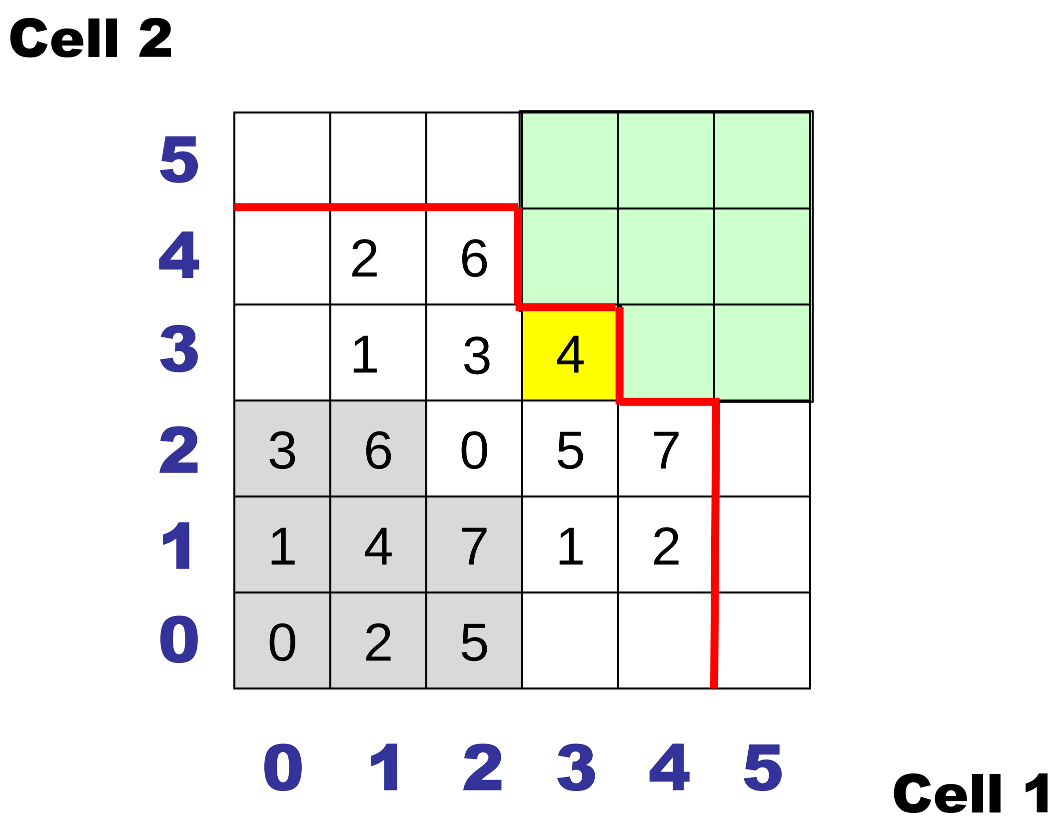
3rd write Sudoku

Cell 2

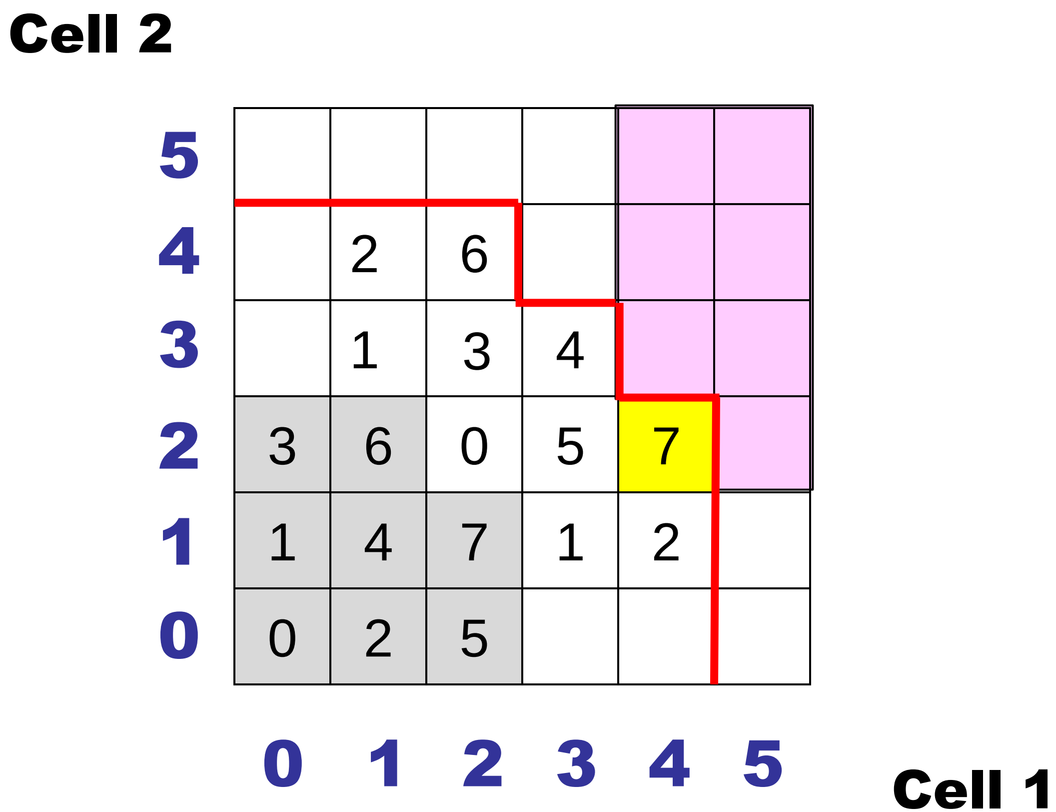
5						
4		2	6			
3		1	3	4		
2	3	6	0	5	7	
1	1	4	7	1	2	
0	0	2	5			
	0	**1**	**2**	**3**	**4**	**5**

Cell 1

3rd write Sudoku



3rd write Sudoku



3rd write Solution

Got t=3!

Cell 2

5			1	7	2	4
4		2	6	0	3	5
3		1	3	4	1	6
2	3	6	0	5	7	0
1	1	4	7	1	2	
0	0	2	5			
	0	1	2	3	4	5

Cell 1

Optimal code family, k=3

Theorem (k=3):

An explicit construction exists with

$$t = \left\lceil \frac{2(q-1)}{3} \right\rceil - 1.$$

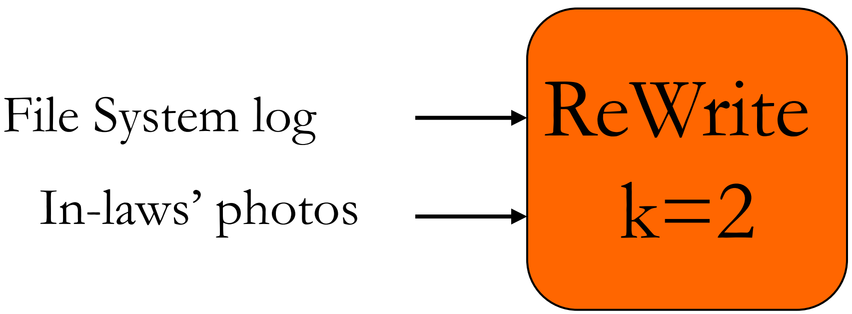
Matching upper bound (k=3):

Any code satisfies

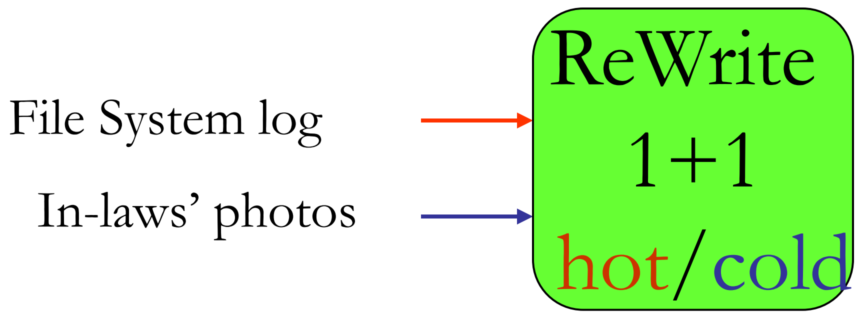
$$t \leq \left\lceil \frac{2(q-1)}{3} \right\rceil - 1.$$

Practical Re-Write Codes II

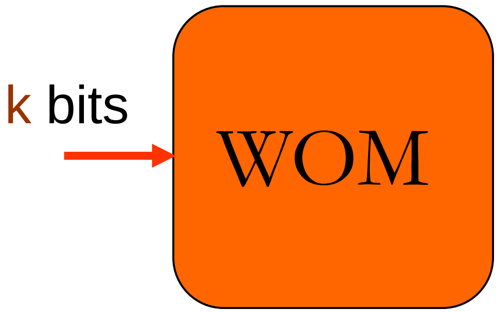
All bits equal



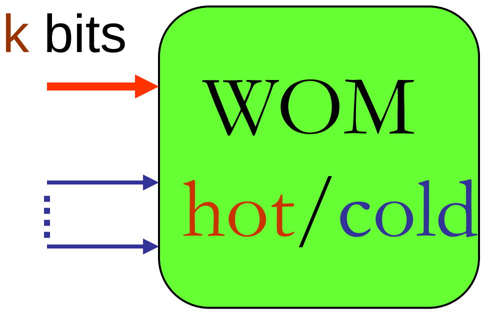
Hot/Cold bits



Hot+Cold Rewrite



t writes



t writes for hot k bits

1 write for each cold bit (anywhere in the write sequence)