Locality and Availability in Distributed Storage

Dimitris Papailiopoulos

Dimacs Workshop on Algorithms for Green Data Storage

joint work with

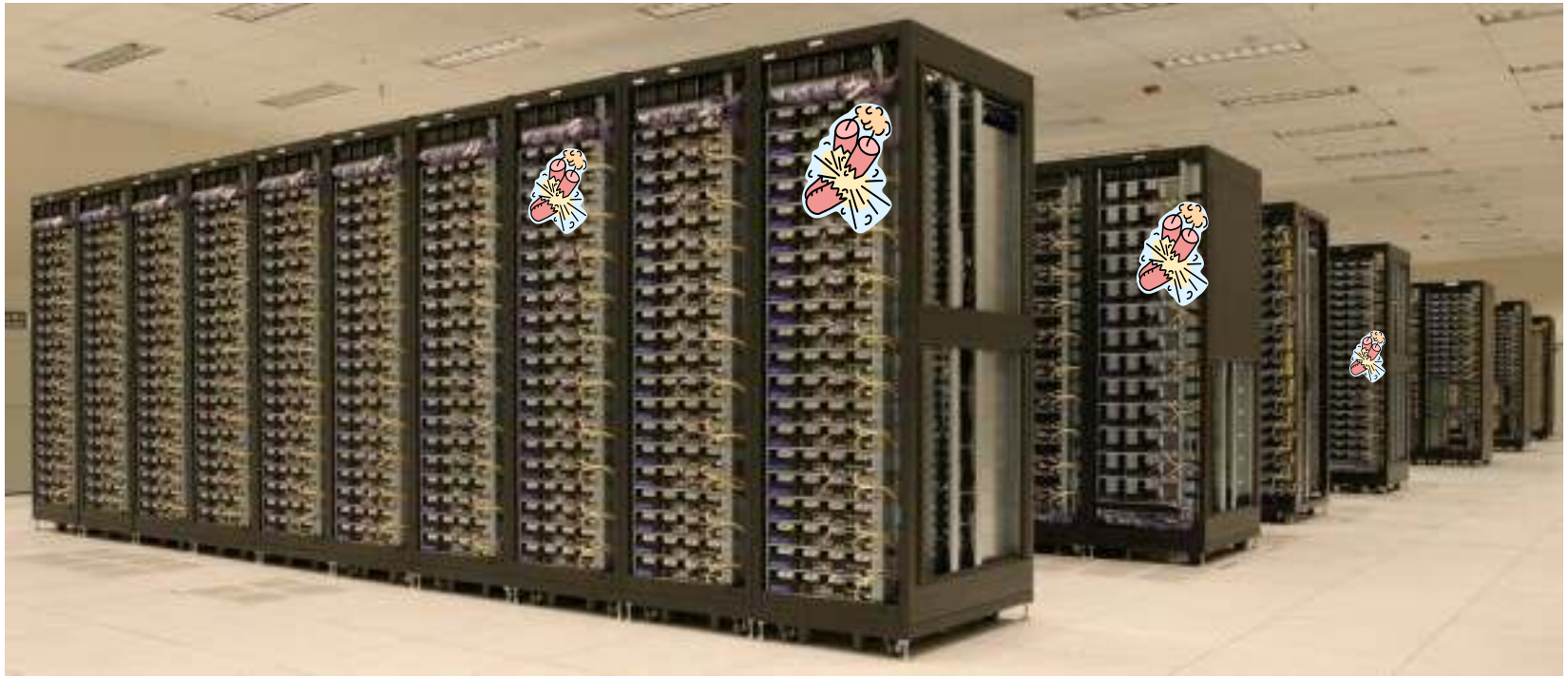
Ankit Rawat Alex Dimakis Sriram Vishwanath

Coding for Distributed Storage

- Current state of the art:
- 3 metrics that measure repair efficiency
- Helping in different system bottlenecks (network vs disk I/O etc).
- Repair locality.
- Mostly coding cold data (rarely accessed)
- (in analytics, most data is cold log data)
- Will define another dimension useful for hot data
- Availability

Reliable Storage

- Large-scale storage (Facebook, Amazon, Google, Yahoo, …)
- FB has the biggest Hadoop cluster (70PB).



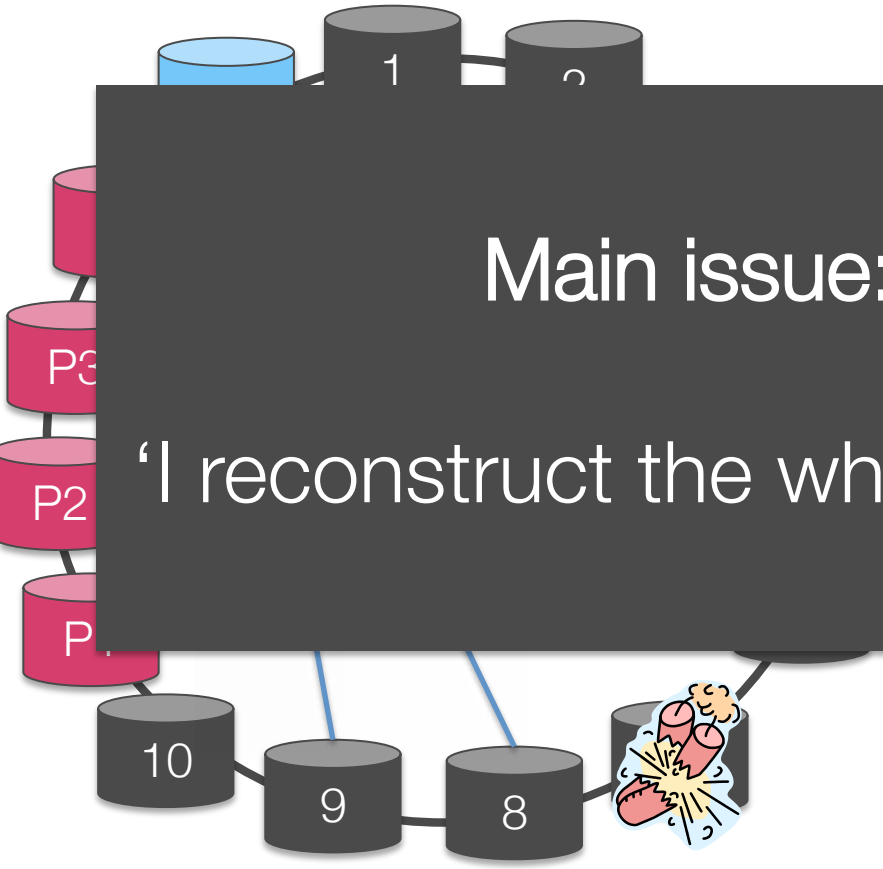
Cluster of machines running Hadoop at Yahoo! (Source: Yahoo!)

- Failures are the **norm.**
- We need to protect the data: **use redundancy**
- **CODING!**

Limitations of Traditional Codes

(14, 10)-RS (fb hdfs raid):

- can tolerate 4 erasures



Main issue: Recovery Cost

‘I reconstruct the whole data to repair 1 node’

1) High network traffic!
2) High disk read!
3) 10x more than the lost information!

Repair Metrics of Interest

- The number of bits communicated during repairs (Repair BW)

  Capacity known (for two extreme points only).

  No high-rate practical codes known for MSR point.

  [Rashmi et al.], [Shah et al.], [El Rouayheb et al.], [Wang et al.], [Tamo et al.], [Suh et al.] [Cadambe et al.] [Papailiopoulos et al.], [Shum], [Oggier et al.] ….

- The number of bits read from disks during repairs (Disk IO)

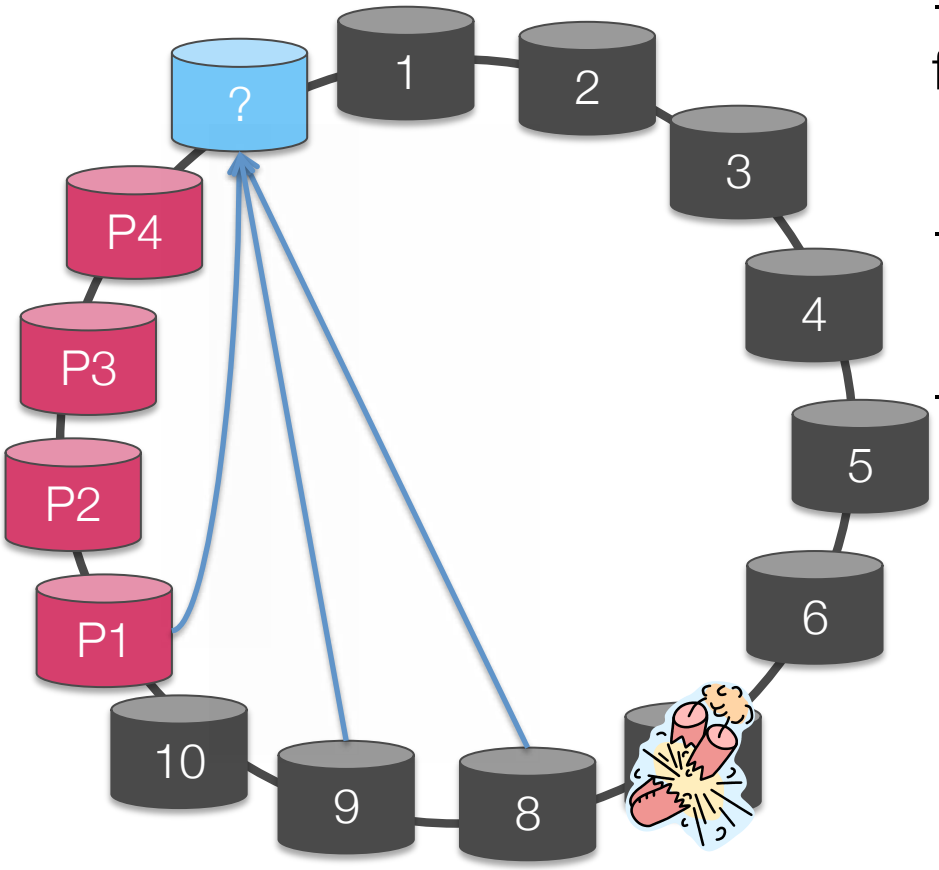
  Capacity unknown

  Only known technique is bounding by Repair Bandwidth

- The number of nodes accessed during a repair (Locality)

  Capacity computed [P, Dimakis, ISIT12, Trans. IT'13].
  Scalar linear bounds [Gopalan et al., Allerton 2011]
  General Code Constructions are open

Low-locality codes?



- A code symbol has locality **r** if it is a function of **r** other codeword symbols.

- Can we have small repair locality?

- And tolerate many erasures (reliability)?

Q: Does locality come at a cost?

Reliability: Minimum Distance

- The distance of a code **d** is the minimum number of erasures after which data is lost.
- Reed-Solomon (10,14) (n=14, k=10). d= 5
- R. Singleton (1964) showed a bound on the best distance possible:

$$d \leq n - k + 1$$

- Reed-Solomon codes achieve the Singleton bound (hence called MDS)

Generalizing Singleton: Locally Repairable Codes

- What happens when we put locality in the picture?

Thm1: an (n,k) code with locality r has

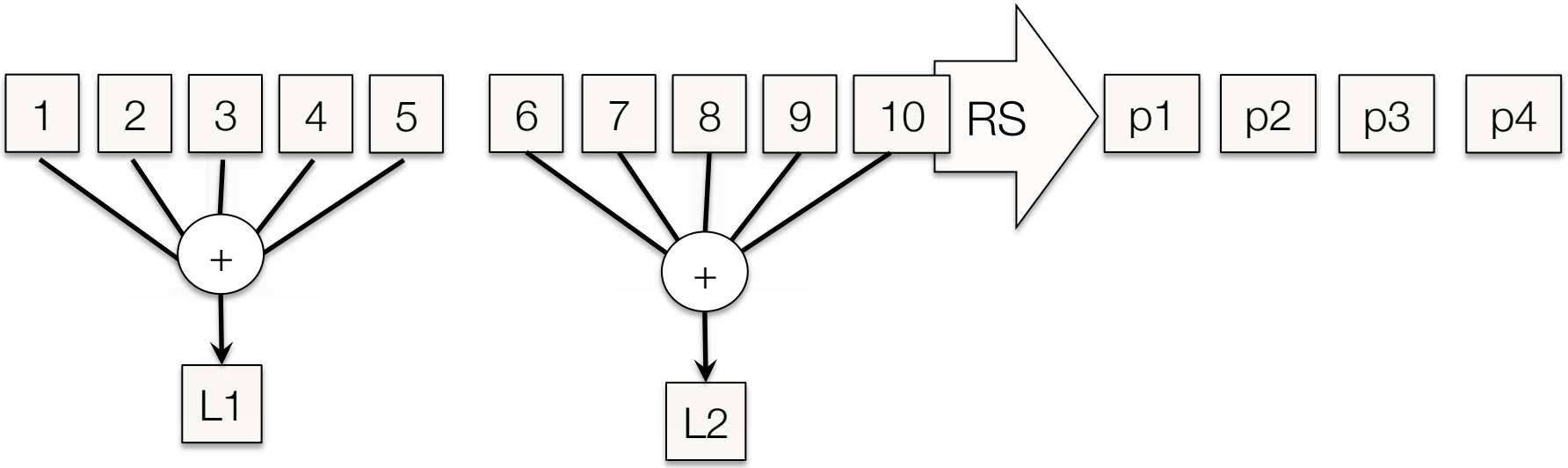
$$d \leq n - k + 1 - \left(\left\lceil \frac{k}{r} \right\rceil - 1 \right)$$

[Gopalan et. al, Allerton11] (scalar-linear codes)
[P., Dimakis, ISIT12, IT13] (information theoretic)

- Non-trivial locality induces a **distance penalty**
- Achievable using random linear network coding [P., Dimakis, ISIT12, IT13]
- Many extensions and explicit constructions

  (Rawat, Silberstein, Tamo, Cadambe, Mazumdar, Forbes…)
- LRCs in MS Azure, they ship with Windows 8.1 [Huang et al. '12]

Example: code with information locality 5

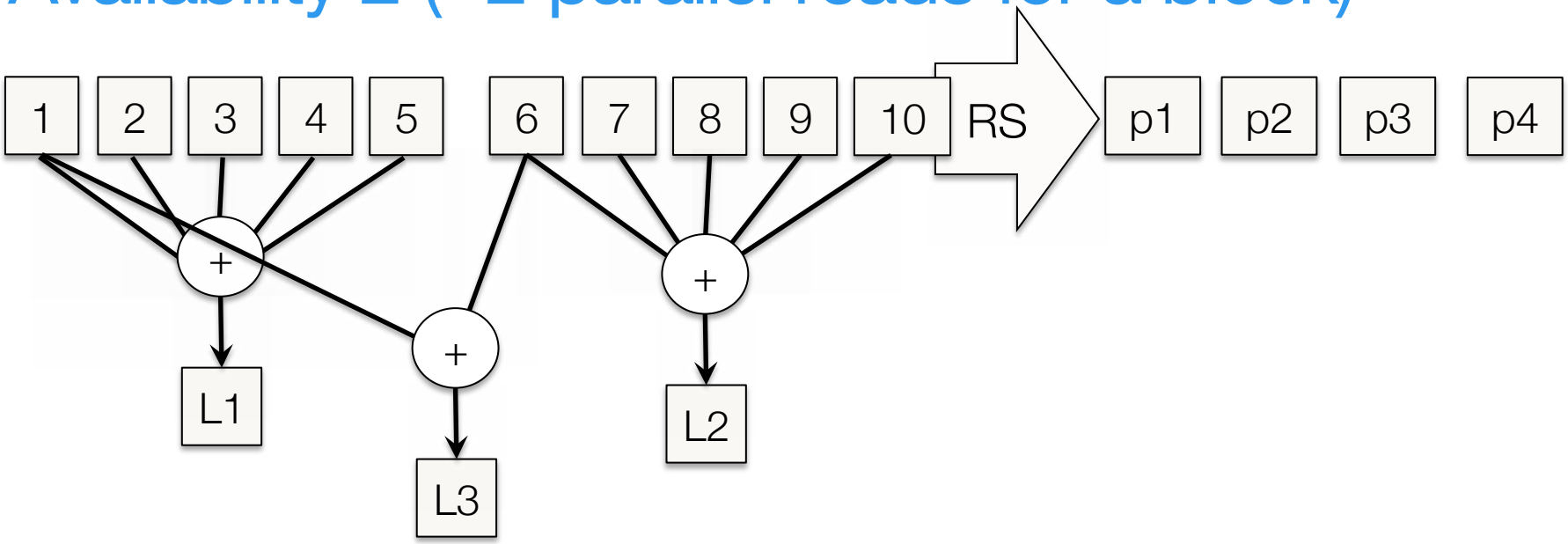


All k=10 message blocks can be recovered by reading r=5 other blocks.

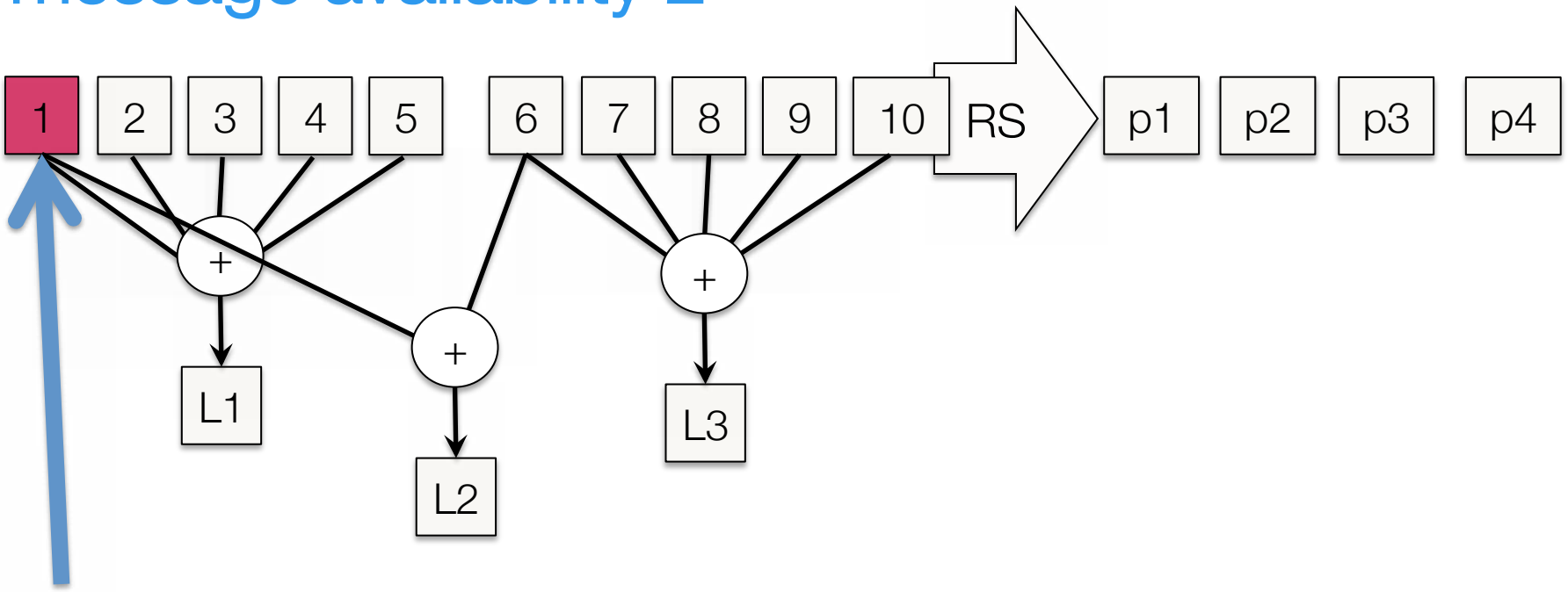
Have to pick L1, L2 in a very structured way (Rawat, Silberstein, Tamo…)

What if I wanted to reconstruct block 1 in parallel?

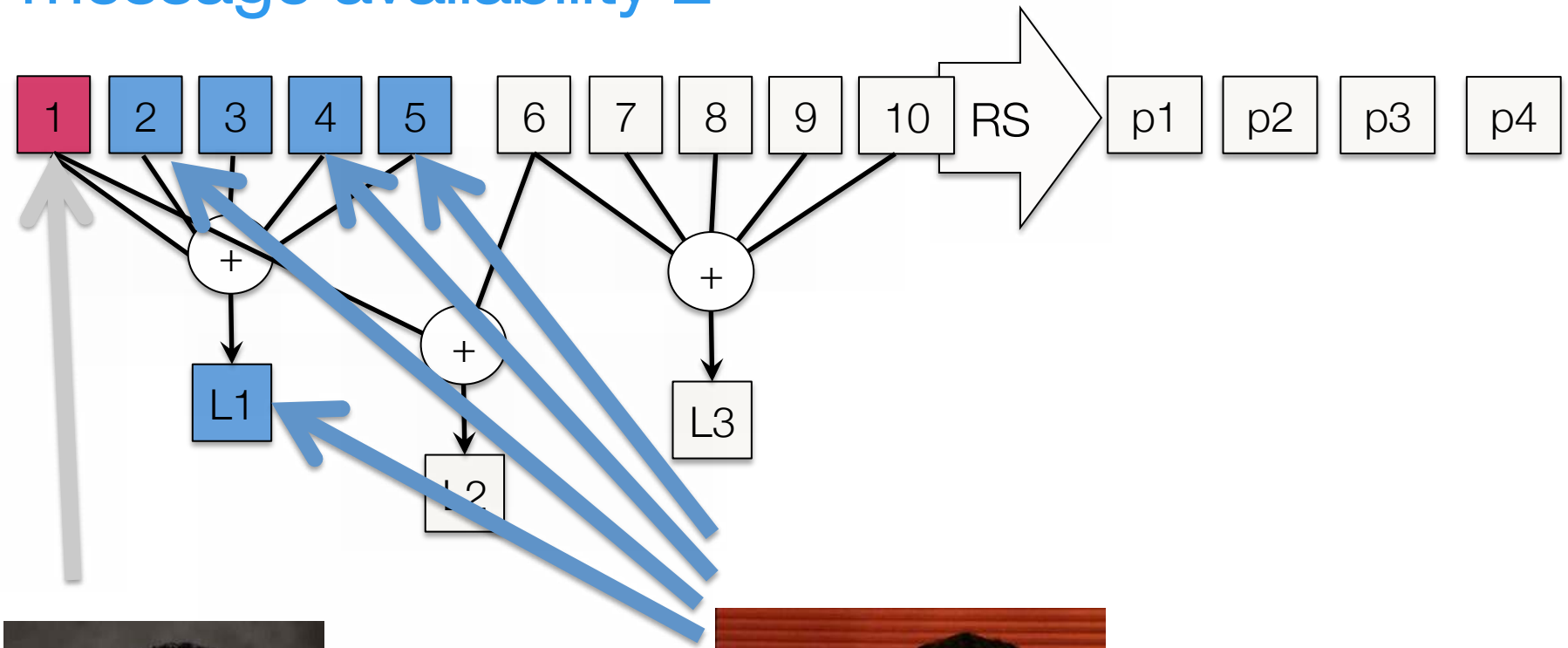
Availability 2 (=2 parallel reads for a block)



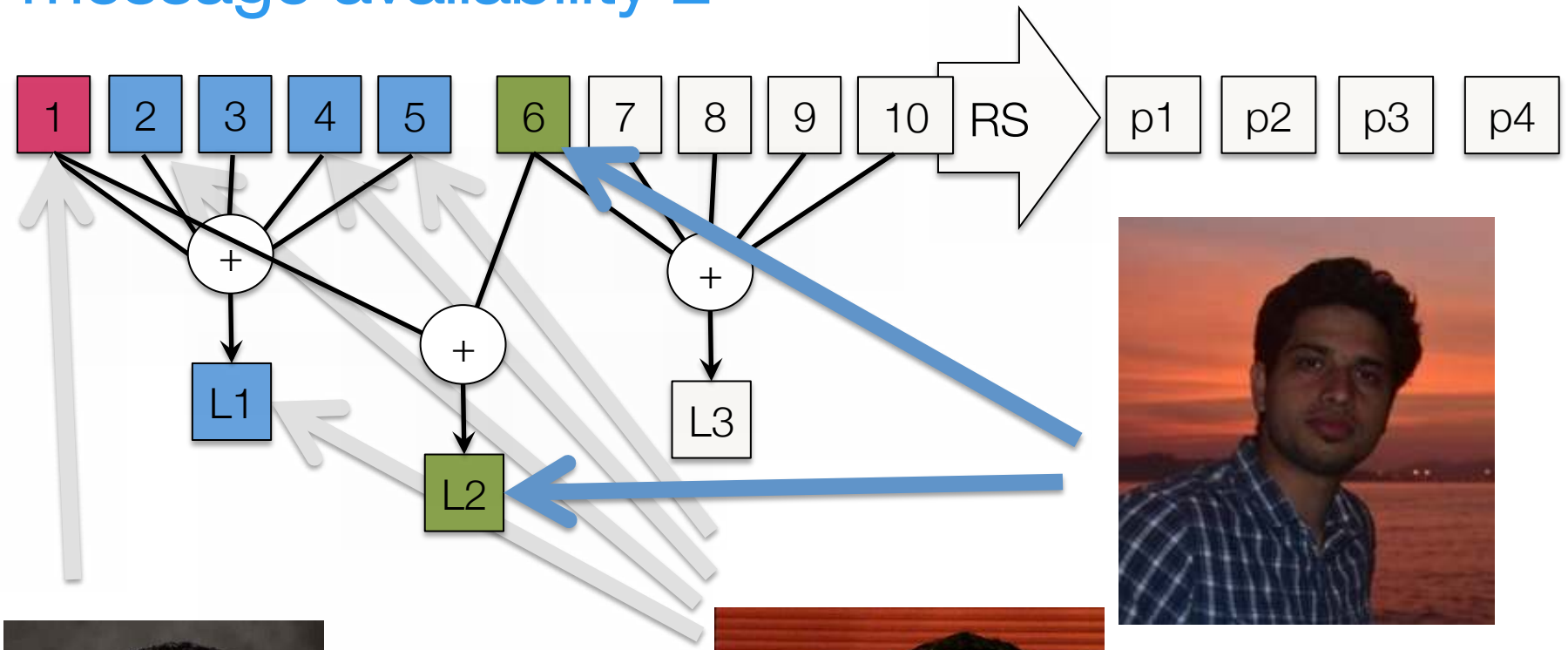
message availability 2



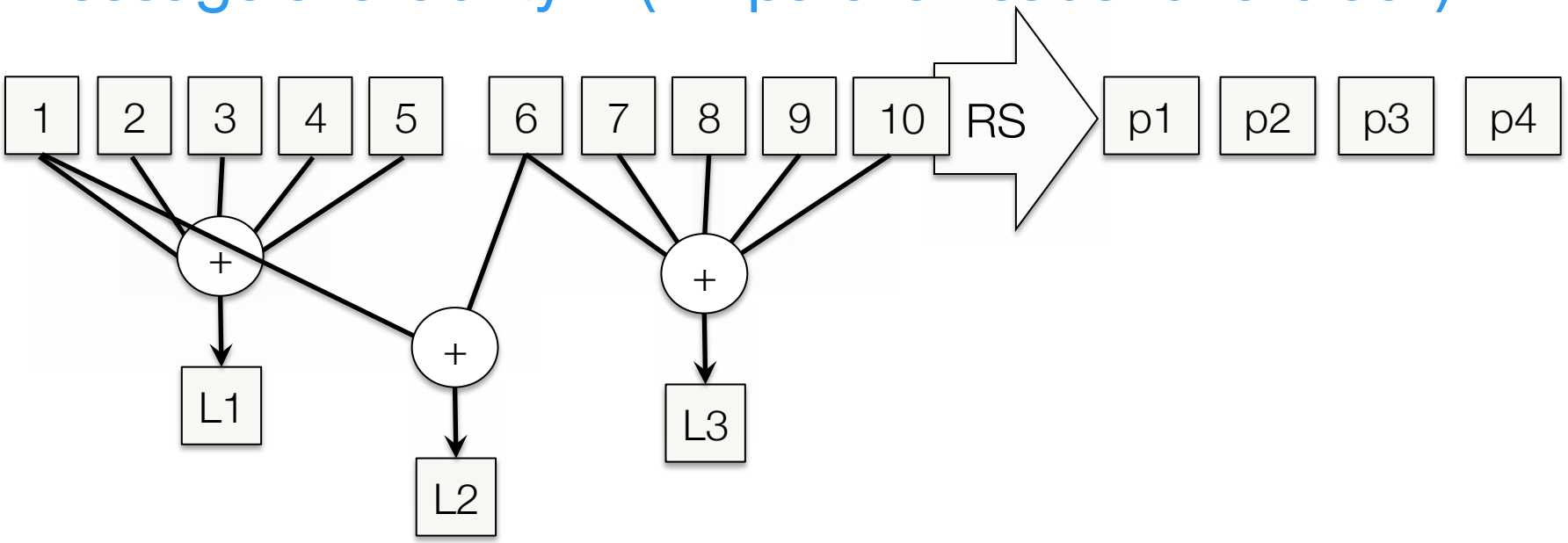
message availability 2



message availability 2



message availability 2 (=2 parallel reads for a block)



- Therefore Block 1 can be read by 1 systematic read + 2 repair reads **simultaneously**
- Block 1 has availability t=2 with groups of locality r1=5 and r2= 2
- Notice also that the group (2,3,4,5,6,7,8,9,10, p1) of locality r=10 can be used to recover 1 (but blocks all others, so not used)

Property: non-overlapping groups of size <= 5

(r, t)-information local code

- For each information (systematic) symbol c_i,
  - t disjoint repair groups.
  - size of each repair group at most r.
- Each systematic symbol has **locality** r and **availability** t.
- **(r, t)-local code:**
  - Code is (r, t)-**information** local code.
  - In addition, non-systematic symbols have locality r.
    - one repair group of size **at most** r.
- (r, 1)-information local code = code with information locality r (MSR LRC)
- (r, 1)-local code = code with all symbol locality r (Facebook LRC)

Q: Does availability come at a cost?

Distance vs. Locality-Availability trade-off

Main Result

- For (r, t)-Information local codes*:

$$d_{\min} \leq n - k + 1 - \left(\left\lceil \frac{kt}{r} \right\rceil - t \right)$$

Distance vs. Locality-Availability trade-off

Main Result

- For (r, t)-Information local codes*:

$$d_{\min} \leq n - k + 1 - \left(\left\lceil \frac{kt}{r} \right\rceil - t \right)$$

*The dirty details:

- We can only prove this for scalar linear codes.
- Only one parity symbol per repair group is assumed.
- Not known what happens for all-symbol availability.
- For some cases we can achieve this using combinatorial designs.

Local Parities using Resolvable Combinatorial Designs

- Set of k symbols: $\mathbf{X} = \{x_1, x_2, \ldots, x_k\}$.
- Family of b subsets (blocks) of X: $\mathbf{B} = \{B_1, B_2, \ldots, B_b\}$.
- $(\mathbf{X}, \mathbf{B})$ is a 2-(k, b, r, c) resolvable design if
  I. $|B_j| = r$ for all $i \in \{1, 2, \ldots, b\}$.
  II. Each symbol appears in c subsets (blocks).
  III. Any two symbols (x_i, x_j) appear in exactly 1 subset (block).
  IV. Design admits **parallelism**:
    - There exist classes $E_1, E_2, \ldots, E_c \subset \mathbf{B}$ such that subsets in E_i partition $\mathbf{X}$.

Property: non-overlapping groups of size = r

Example[1]

- 2-(k, b, r, c) = 2-(15, 35, 3, 7) resolvable design.

$\{1,2,3\}$	$\{1,5,6\}$	$\{2,6,11\}$	$\{2,5,9\}$	$\{3,5,11\}$	$\{5,7,13\}$	$\{3,4,13\}$
$\{6,7,8\}$	$\{4,7,11\}$	$\{4,5,12\}$	$\{3,7,12\}$	$\{2,4,8\}$	$\{2,12,14\}$	$\{5,8,14\}$
$\{11,12,13\}$	$\{8,9,12\}$	$\{9,10,13\}$	$\{1,8,13\}$	$\{7,9,15\}$	$\{3,6,9\}$	$\{6,12,15\}$
$\{4,9,14\}$	$\{3,10,14\}$	$\{3,8,15\}$	$\{11,14,15\}$	$\{1,10,12\}$	$\{8,10,11\}$	$\{1,9,11\}$
$\{5,10,15\}$	$\{2,13,15\}$	$\{1,7,14\}$	$\{4,6,10\}$	$\{6,13,14\}$	$\{1,4,15\}$	$\{2,7,10\}$

[1] Kirkman's schoolgirl problem: 15 girls walking in groups of 3, each day of the week. How to place them so that no two walk twice together.

Proposed by Rev. Thomas Kirkman in 1850.

The first solution was by Arthur Cayley. This was shortly followed by Kirkman's own solution. J.J. Sylvester also investigated the problem and ended up declaring that Kirkman stole the idea from him.

Example[1]

- 2-(k, b, r, c) = 2-(15, 35, 3, 7) resolvable design.

Subset (block)

$\{1, 2, 3\}$	$\{1, 5, 6\}$	$\{2, 6, 11\}$	$\{2, 5, 9\}$	$\{3, 5, 11\}$	$\{5, 7, 13\}$	$\{3, 4, 13\}$
$\{6, 7, 8\}$	$\{4, 7, 11\}$	$\{4, 5, 12\}$	$\{3, 7, 12\}$	$\{2, 4, 8\}$	$\{2, 12, 14\}$	$\{5, 8, 14\}$
$\{11, 12, 13\}$	$\{8, 9, 12\}$	$\{9, 10, 13\}$	$\{1, 8, 13\}$	$\{7, 9, 15\}$	$\{3, 6, 9\}$	$\{6, 12, 15\}$
$\{4, 9, 14\}$	$\{3, 10, 14\}$	$\{3, 8, 15\}$	$\{11, 14, 15\}$	$\{1, 10, 12\}$	$\{8, 10, 11\}$	$\{1, 9, 11\}$
$\{5, 10, 15\}$	$\{2, 13, 15\}$	$\{1, 7, 14\}$	$\{4, 6, 10\}$	$\{6, 13, 14\}$	$\{1, 4, 15\}$	$\{2, 7, 10\}$

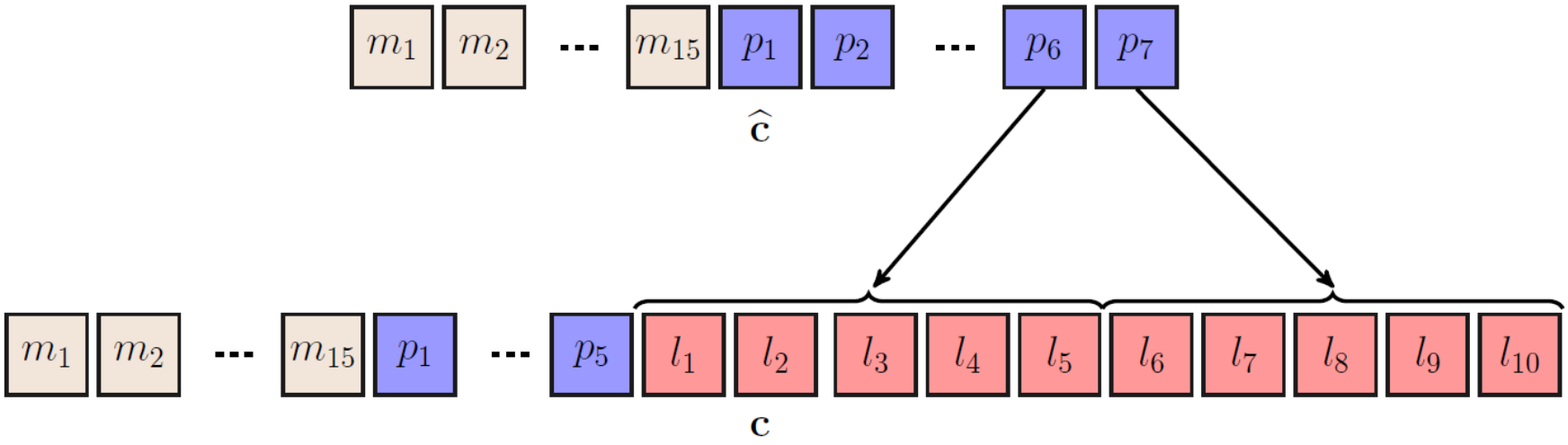
Class

- Subsets (blocks) in each class (column) partition set X = {1, 2,…, 15).

[1] Kirkman's schoolgirl problem: http://en.wikipedia.org/wiki/Kirkman%27s_schoolgirl_problem.

Example

- (n,k, r, t) = (30, 15, 3, 2) and N = 20.



- First **two (t = 2)** classes of the resolvable design from Kirkman's schoolgirl problem are used to split p_6 and p_7.

$l_1 = a_1m_1 + a_2m_2 + a_3m_3$	$l_6 = b_1m_1 + b_5m_5 + b_6m_6$
$l_2 = a_6m_6 + a_7m_7 + a_8m_8$	$l_7 = b_4m_4 + b_7m_7 + b_{11}m_{11}$
$l_3 = a_{11}m_{11} + a_{12}m_{12} + a_{13}b_{13}$	$l_8 = b_8m_8 + b_9m_9 + b_{12}m_{12}$
$l_4 = a_4m_4 + a_9m_9 + a_{14}m_{14}$	$l_9 = b_3m_3 + b_{10}m_{10} + b_{14}m_{14}$
$l_5 = a_5m_5 + a_{10}m_{10} + a_{15}m_{15}$	$l_{10} = b_2m_2 + b_{13}m_{13} + b_{15}m_{15}$

Conclusions

- Locality–Distance Trade-off
- Defined **Availability**: the number of parallel reads allowed by a code.
- Showed a tradeoff between distance-locality and availability.
- Created codes with good availability using combinatorial designs.
- All-symbol availability remains open as well as vector-linear codes.
- Also achievability remains open in many cases.