Software-based Erasure Codes for Scalable Distributed Storage

J. Cooley, J. Mineweaser, L. Servi, E. Tsung

20th IEEE Mass Storage Systems Symposium

11th NASA Goddard Mass Storage Systems and Technologies Conference

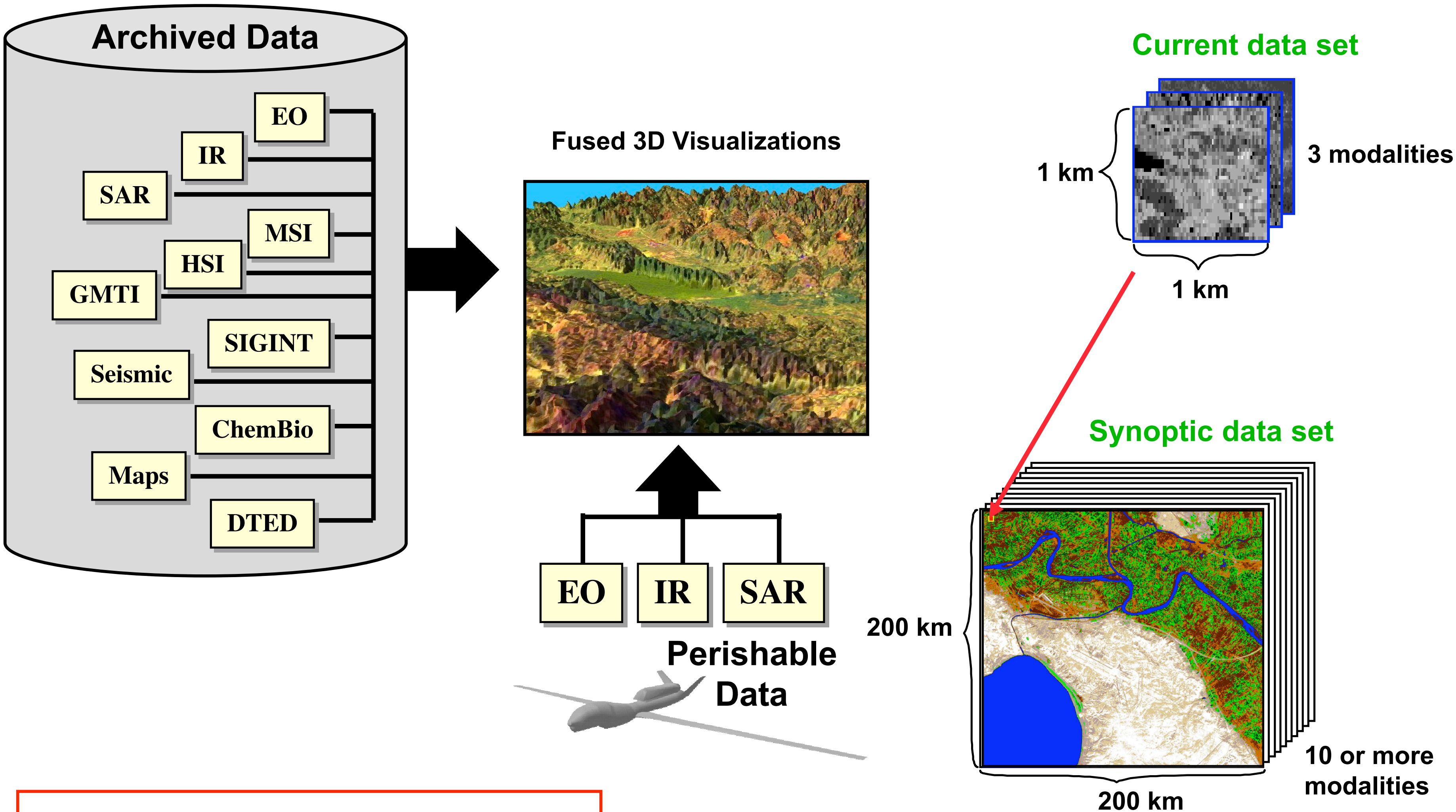
This work is sponsored by the United States Air Force under Air Force Contract #F19628-00-C-0002. Opinions, interpretations, recommendations and conclusions are those of the authors and are not necessarily endorsed by the United States Government.

April 2003

Outline

- General motivation for distributed storage systems
- Approaches for protecting data using codes
- Implementation issues for graph-based codes
  - Encoding/Decoding with graph-based codes
  - Luby Transform (LT) Code
  - Lincoln Erasure Code (LEC)
- Experiments
  - Throughput comparison
  - Reliability comparison
- Summary

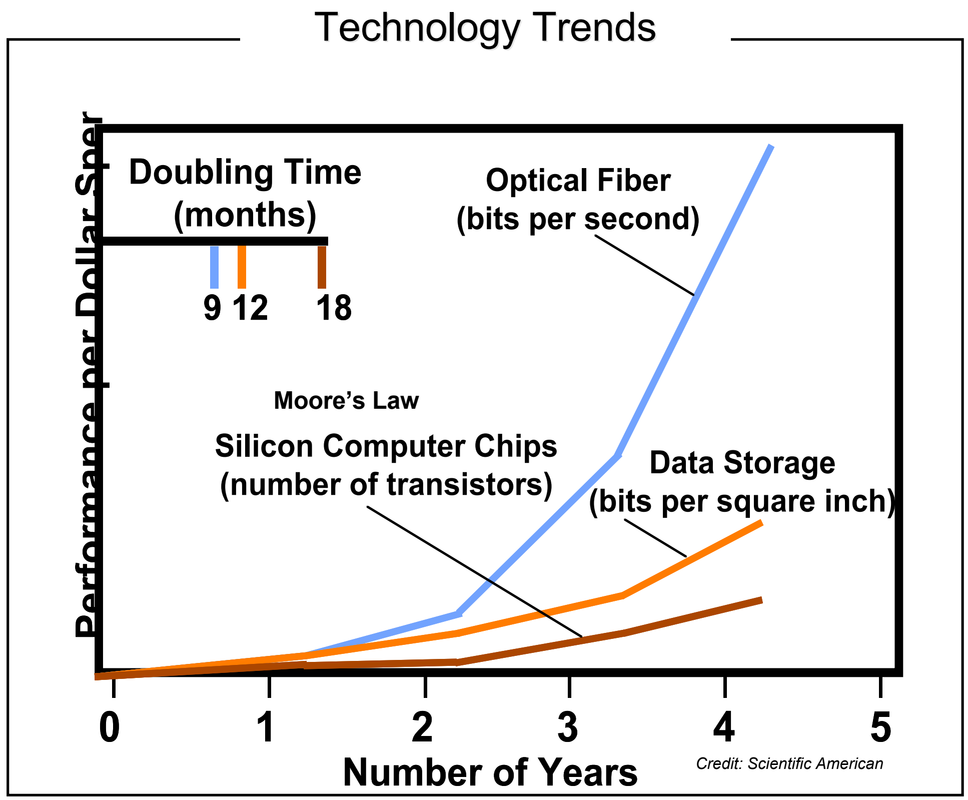
The Sensor Data Explosion



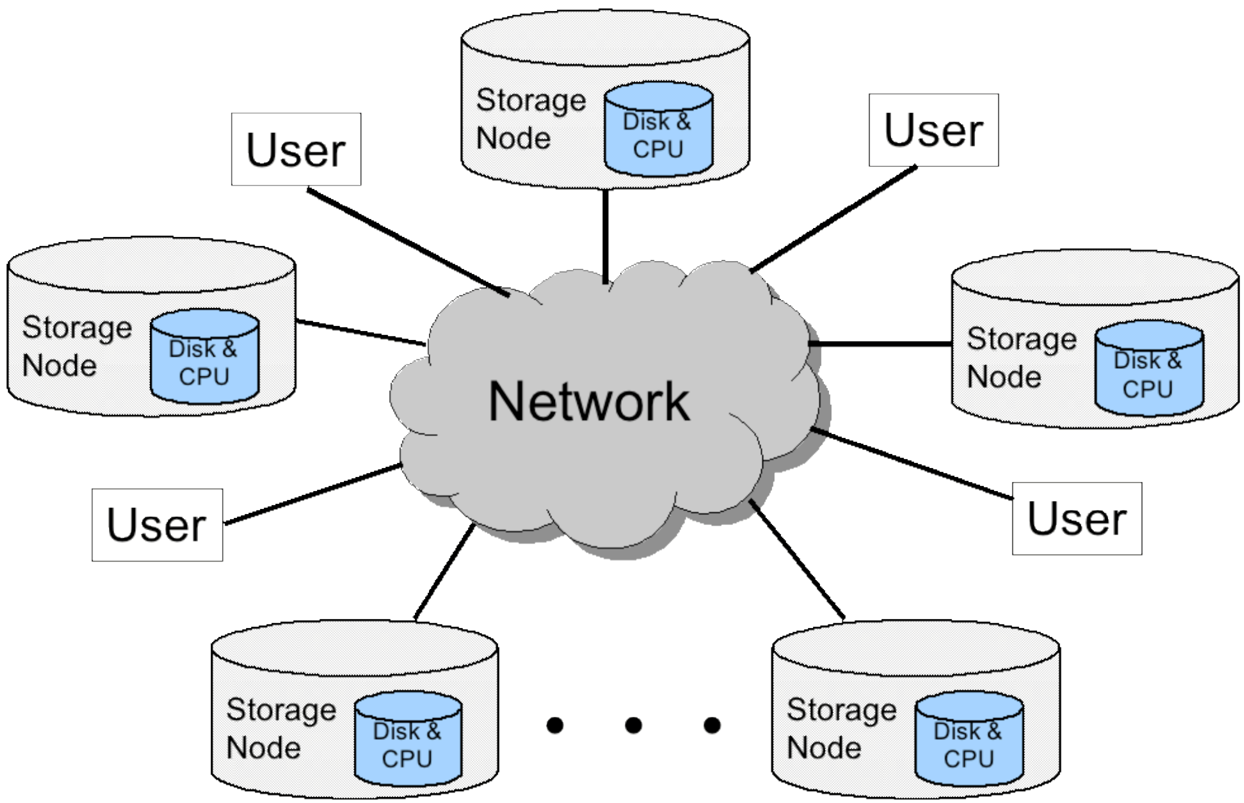
Data growth > 120,000X

Critical Technology Trends

- Follow the commercial technology trends in
  - Networking
  - Data Storage
  - Processing
- Capabilities in 2002
  - 10 Gbps Networks
  - 160 Gigabyte Disks
- Estimates in ~ 2010
  - 10 Tbps Networks
  - 40 Terabyte Disks

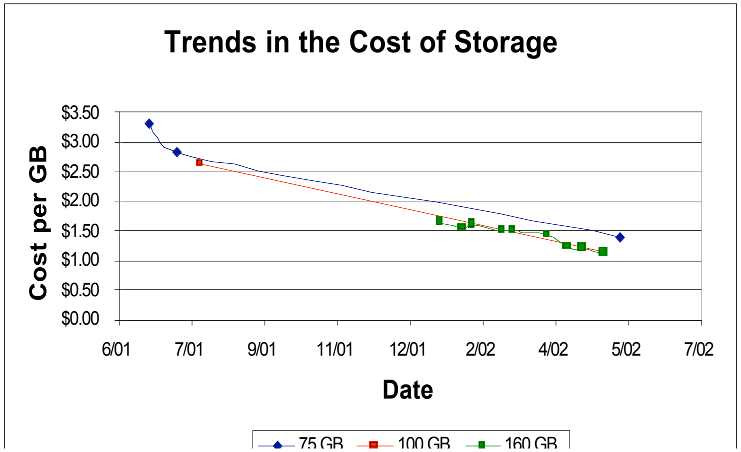


Distributed Storage System Architecture



Hardware for Distributed Storage

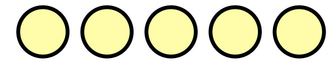
- Commercial-off-the-shelf (COTS) Components
  - Leverage economies of scale
  - Interoperable open standards
  - In the last year …
    - Storage density has doubled …
    - … and cost per GB is down 50%



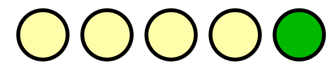
- Custom System Integration
  - Aggregate low-cost storage nodes
  - Matched to unique user requirements
  - Performance can be scaled over time

Erasure Coding for Data Storage

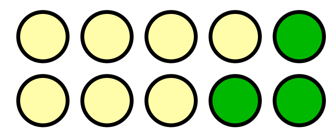
- **Replication** — (N,1)
  - No encoding, decoding required
  - Negative impact on available storage capacity



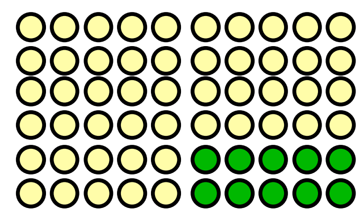
- **Single Parity Check** — (N+1,N)
  - Simple encoding and decoding
  - Protects only against a single failure



- **Reed-Solomon** — (N,K)
  - Minimum distance, but computationally intensive
  - Commercial hardware limited to small blocks
  - Software implementations are relatively slow

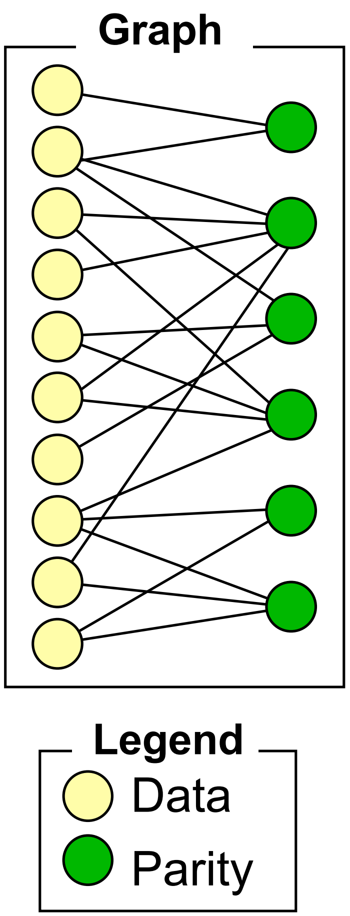


- **Graph-based Low-density Parity Check** — (N,K)
  - Minimum distance traded for high performance
  - Software implementations are sufficiently fast
  - Scales to large block sizes (up to thousands)

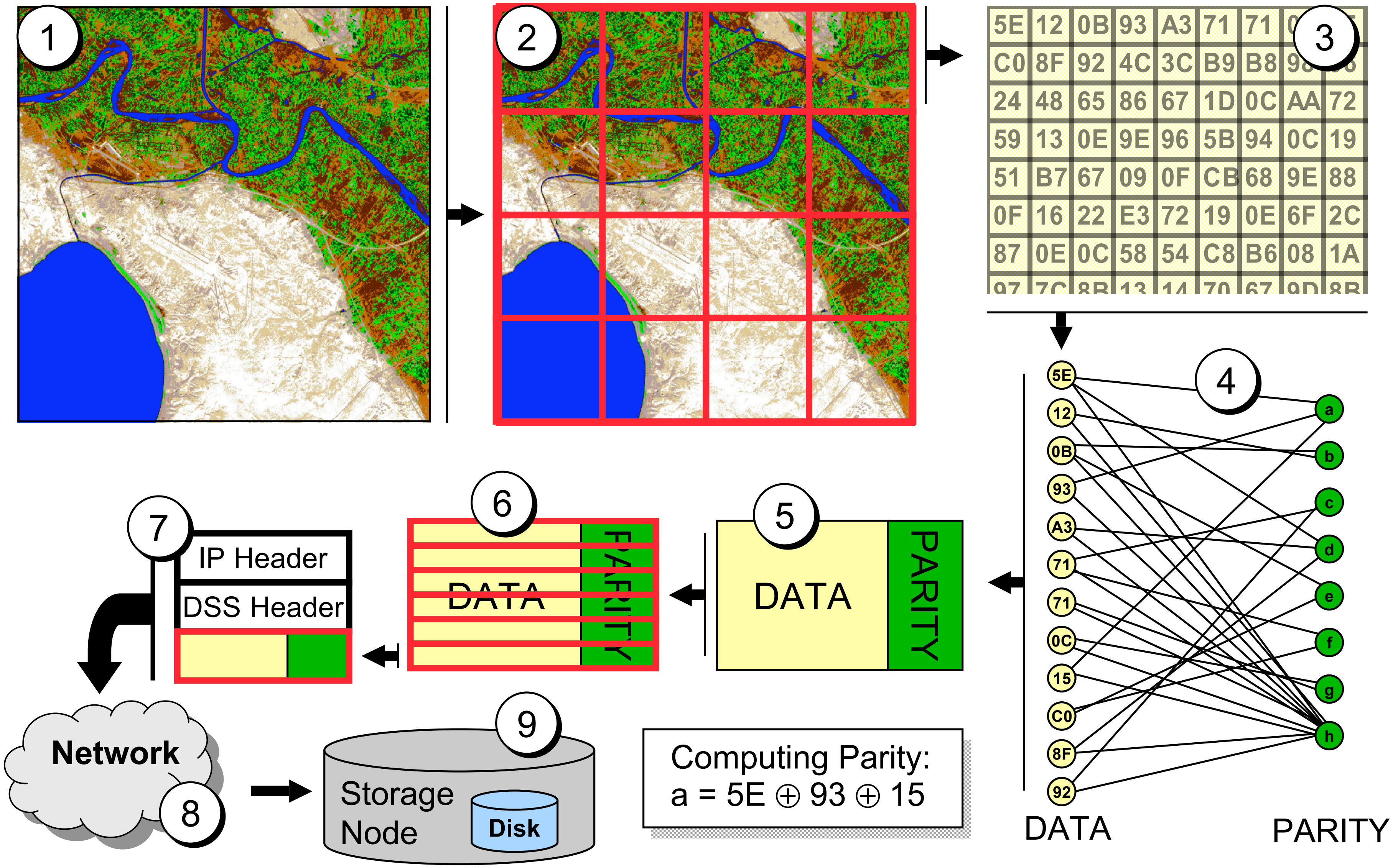


Graph-based, Large Block Erasure Coding

- Codes on Graphs
  - Large blocks, low density
  - Bipartite: data, parity nodes
  - Heavy-tail degree distribution
  - Random assignment of edges
- Simple serial/parallel encoder
  - Steady throughput of 100's Mbps
- Iterative decoder
  - Throughput depends on erasures
  - 100's Mbps with maximum losses
- Probabilistic erasure correction
  - Tune to meet user requirements
  - ~ 5% overhead for 10^{-9} reliability



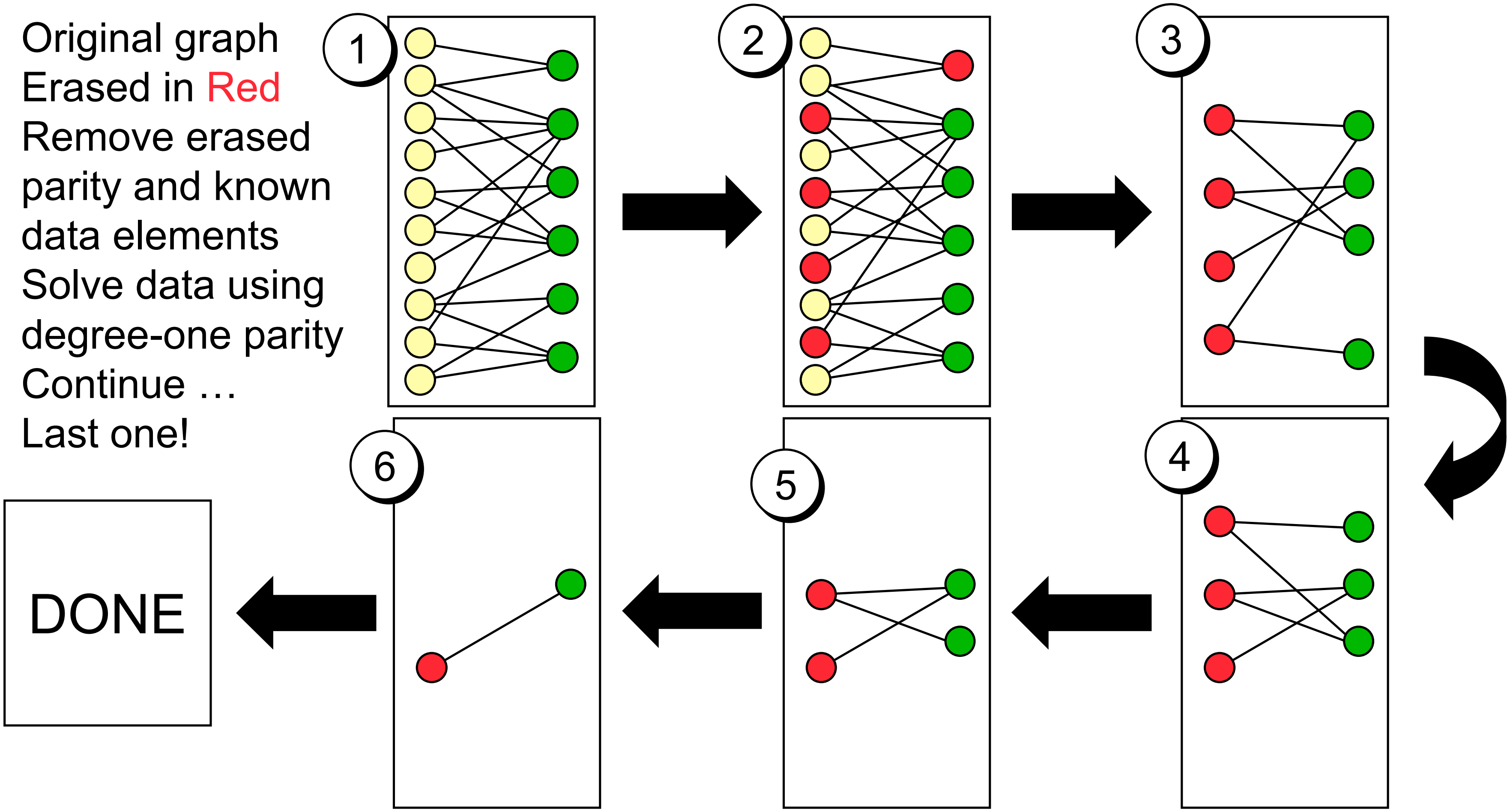
Encoding Data for Distributed Storage



An Iterative Decoder for Erasure Codes

- **Draw graph, including erased nodes and adjacent nodes**
- **For each parity node of degree one, correct erased node**

1. Original graph
2. Erased in Red
3. Remove erased parity and known data elements
4. Solve data using degree-one parity
5. Continue …
6. Last one!

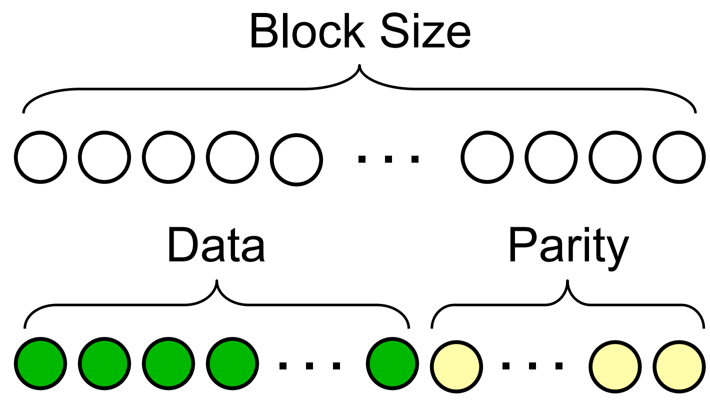


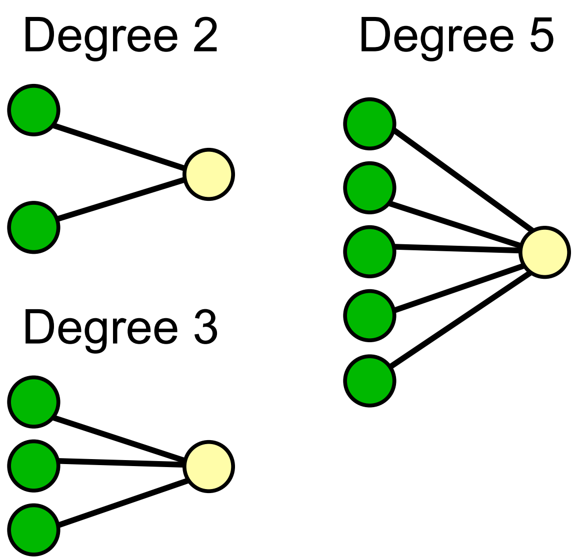
Choosing a Graph Code

- **Specify the Block Size**
  - **7500 Elements**
- **Define Data/Parity Split**
  - **5000 Data**
  - **2500 Parity**
- **Parity Degree Distribution**

Degree	Quantity
2	1100
3	600
4	300
5	200
…	…
3000	1

- **Edge Assignment**
  - **For parity of degree *d*, randomly select subset of *d* data elements**





Luby Transform (LT) Code

- Similarity with our approach
  - High performance graph-based coding approach

- Differences
  - Different motivation: LT sought reliable broadcast capability
  - Bits sent until file can be reconstructed
  - Sent no data bits
    - Good for broadcast application
    - Slower for storage application

Lincoln Erasure Code (LEC)

The number of parity nodes of degree j is given by the following equation:

$$N_j(s,p,k,A,j^*) = s \square \square \frac{Lk}{j \square j^* \square 1} \square \frac{Lk + 1 \square A\sqrt{Lk}}{j \square \square} \square$$

for j = j* + 2, j* + 3, …, imax

where s is a scaling factor, p is the fraction of corruption being protected against, k is the number of data nodes, A is a variable, and L is the desired loss protection.

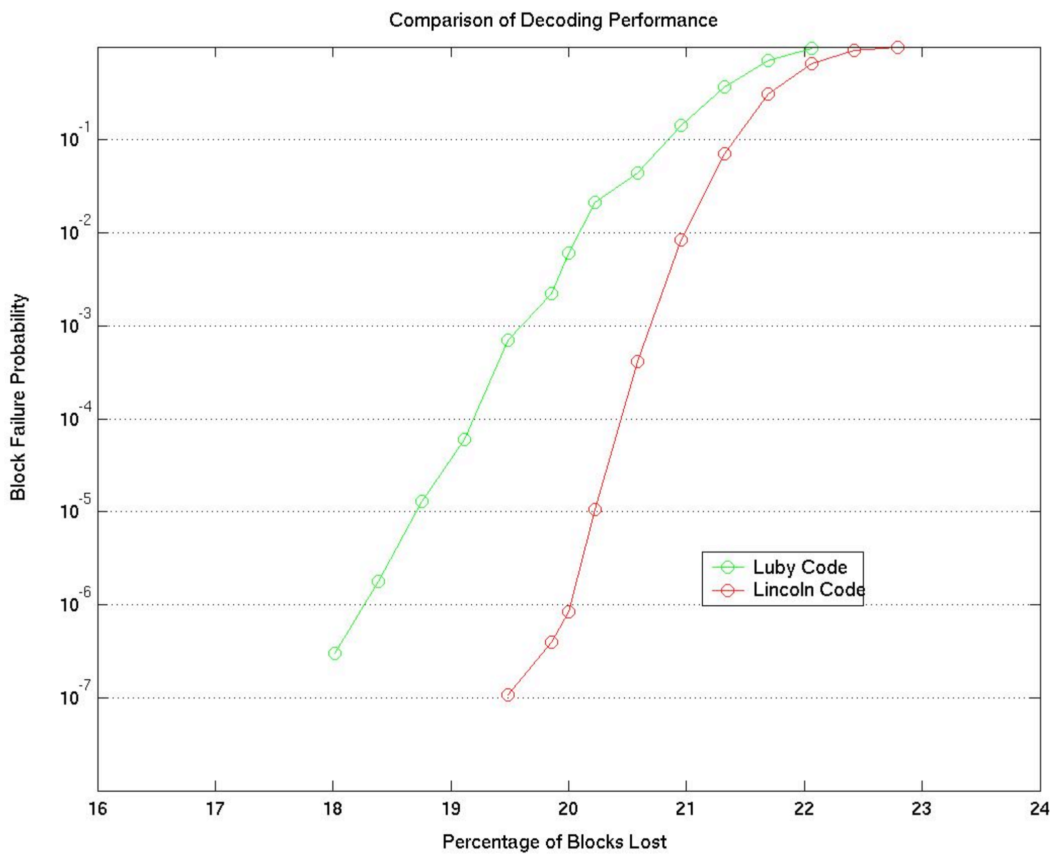
Experimental Plan

- Thoughput comparison
- Reliability comparison
  - File size: 13.1 Mbyte
  - Loss rates
    - 1%
    - 10 %
    - 20 %
    - 50 %
    - 75 %
  - Block size
    - 5040 data
    - 2520 data
  - Targeted probability of unsuccessful file reconstruction
    - 10^{-6}
    - 10^{-4}

Throughput Comparison of Erasure Codes

Erasure Coding Method	Encoding (Mbps)	Decoding (20% Loss) (Mbps)
LEC	240	760
Luby Code	180	220
Reed-Solomon	0.09	0.20

Reliability Comparison of Erasure Codes



Block of 5040 data and 1760 parity elements

Comparison of Reliability with Various Percent Loss and 5040 Data

5040 data, targeting 10^{-6} reliability

Percent Loss	# Parity Elements	LEC Reliability	Luby Reliability	Minimum # Parity Elements Needed	Overhead
1	130	6.5×10^{-7}	1.0	51	2.55
10	860	3.3×10^{-7}	6.1×10^{-1}	560	1.54
20	1760	8.3×10^{-7}	6.1×10^{-3}	1260	1.40
50	6350	1.7×10^{-6}	3.5×10^{-6}	5040	1.26
75	20000	2.2×10^{-6}	2.8×10^{-7}	15120	1.42

5040 data, targeting 10^{-4} reliability

Percent Loss	# Parity Elements	LEC Reliability	Luby Reliability	Minimum # Parity Elements Needed	Overhead
1	110	1.4×10^{-4}	1.0	51	2.16
10	810	2.3×10^{-5}	1.0	560	1.45
20	1700	3.5×10^{-5}	2.9×10^{-2}	1260	1.35
50	6230	9.9×10^{-5}	1.7×10^{-5}	5040	1.24
75	19250	1.1×10^{-4}	7.5×10^{-7}	15120	1.27

Comparison of Reliability with Various Percent Loss and 2520 Data

2520 data, targeting 10^{-6} reliability

Percent Loss	# Parity Elements	LEC Reliability	Luby Reliability	Minimum # Parity Elements Needed	Overhead
1	80	1.9×10^{-6}	1.0	25	3.20
10	470	2.3×10^{-6}	9.0×10^{-1}	280	1.68
20	950	1.0×10^{-6}	3.2×10^{-2}	630	1.51
50	3360	7.3×10^{-7}	6.8×10^{-6}	2520	1.33
75	10400	1.8×10^{-6}	1.1×10^{-6}	7560	1.37

2520 data, targeting 10^{-4} reliability

Percent Loss	# Parity Elements	LEC Reliability	Luby Reliability	Minimum # Parity Elements Needed	Overhead
1	65	1.4×10^{-4}	1.0	25	2.60
10	435	1.4×10^{-4}	1.0	280	1.55
20	900	5.8×10^{-5}	2.9×10^{-1}	630	1.43
50	3240	3.6×10^{-5}	2.6×10^{-4}	2520	1.29
75	10100	2.0×10^{-4}	1.8×10^{-6}	7560	1.34

Summary

- **Motivated need for codes in storage systems**
- **Introduced Lincoln Erasure Codes (LEC)**
  - **Defined**
  - **Described implementation of code in storage systems**
  - **Demonstrated throughput and reliability performance**