Erasure-Coded Key-Value Stores with Side Information

Ramy E. Ali

Algorithms, Analytics & Augmented Intelligence group, Math of Communications department, August 2018

NOKIA Bell Labs

PennState College of Engineering | ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

Outline

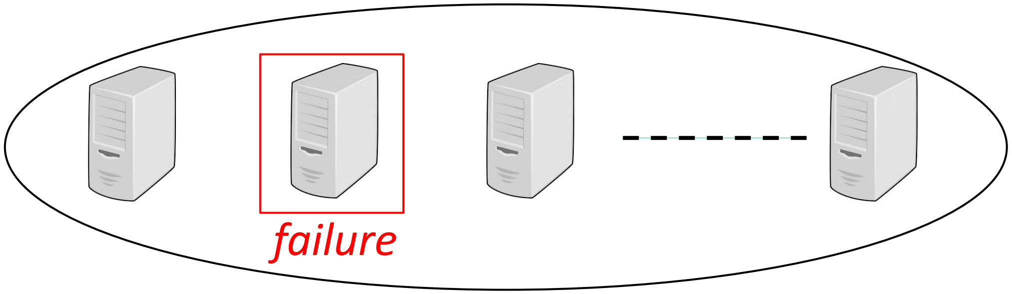
- Key-value Stores Overview
- Background: Replication & Erasure Coding
- Coding with Side Information: Problem Formulation
- Impossibility Results
- Code Constructions
- Case Study: Latency-Storage Trade-off in AWS
- Discussion

Key-value Stores

- Applications: reservation systems, financial transactions, distributed computing, …
- Numerous key-value stores: Amazon Dynamo, Apache Cassandra, and CouchDB

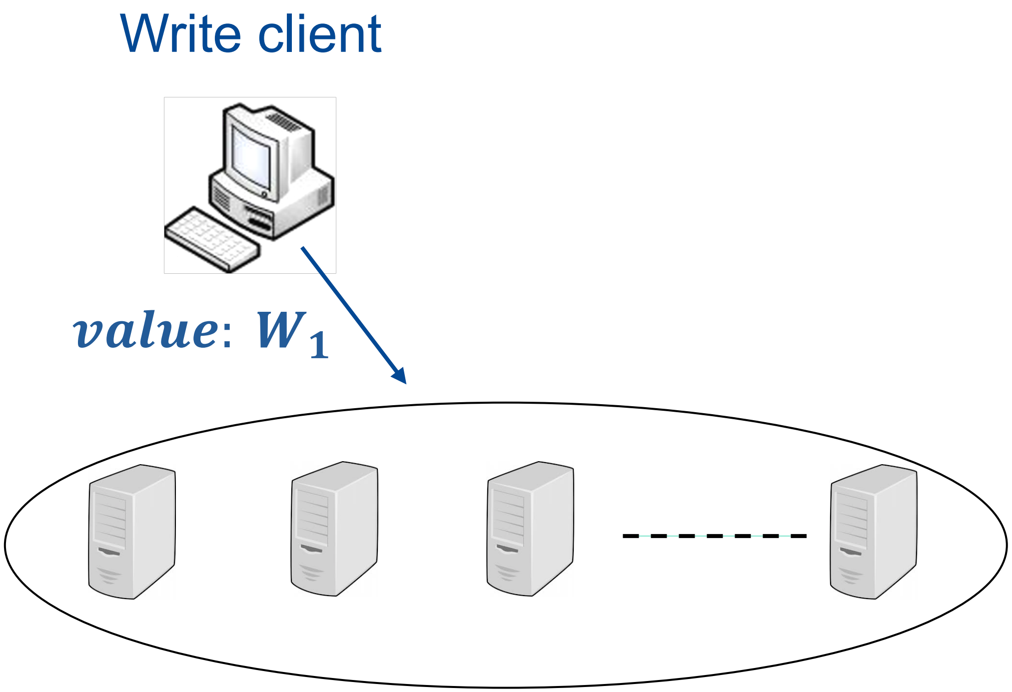
Distributed Key-value Stores

- Data is stored over multiple nodes.



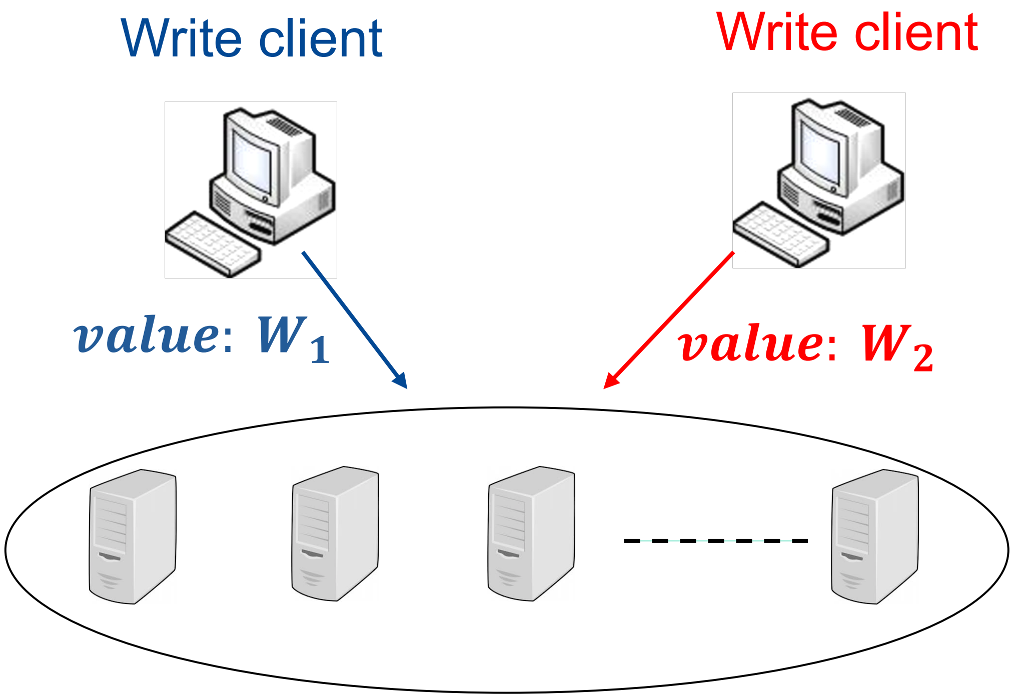
Distributed Key-value Stores

- Data is stored over multiple nodes.
- Data is asynchronously updated.



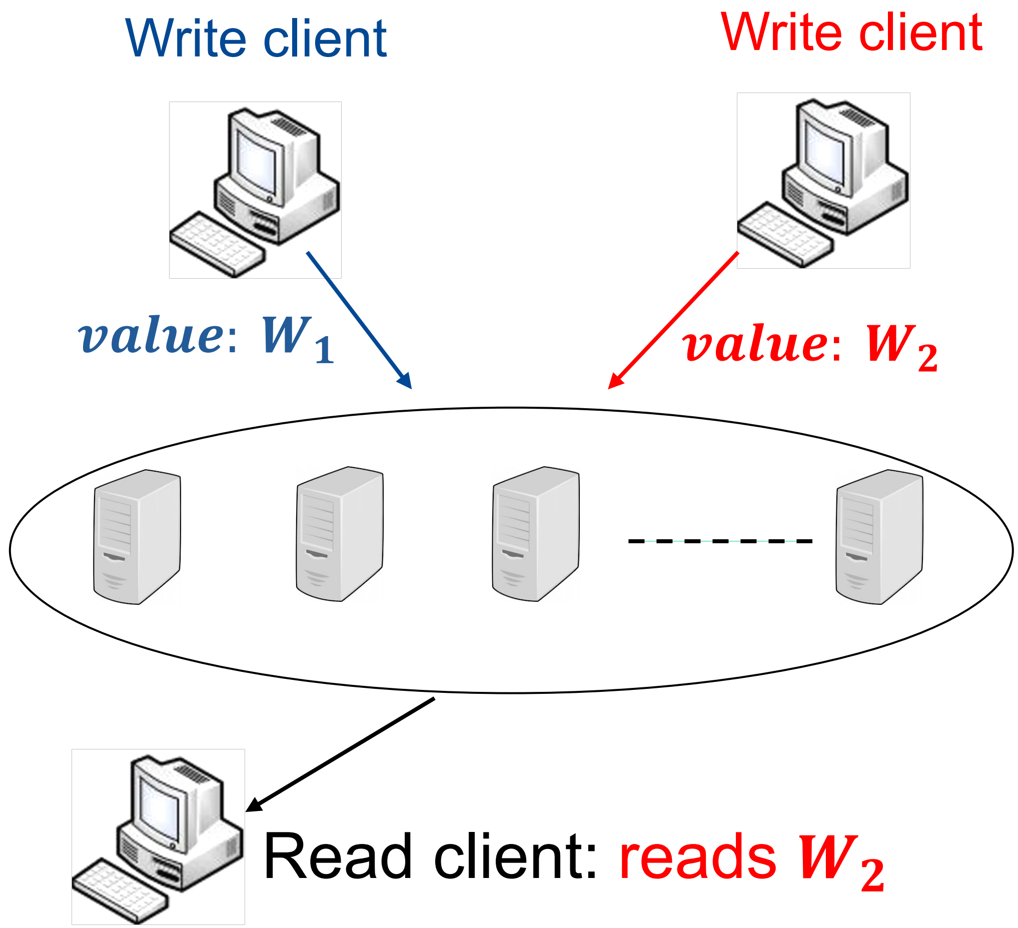
Distributed Key-value Stores

- Data is stored over multiple nodes.
- Data is asynchronously updated.



Distributed Key-value Stores

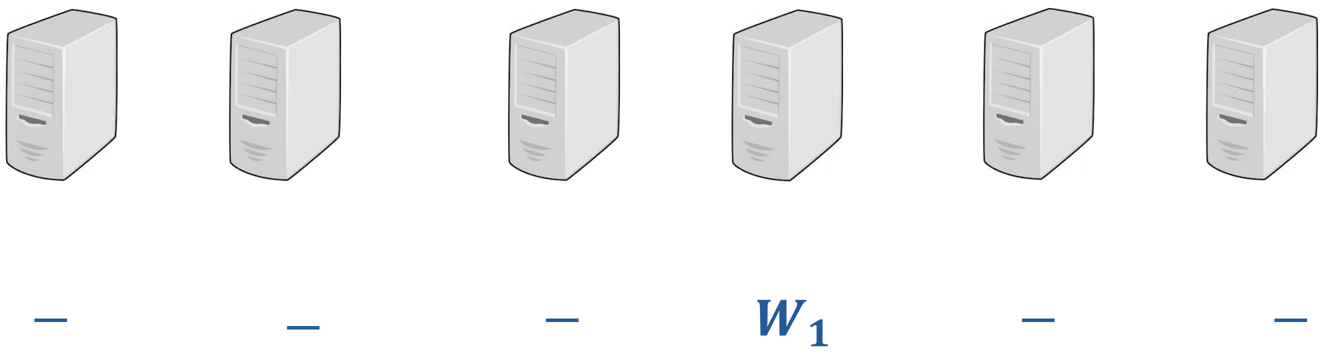
- Data is stored over multiple nodes.
- Data is asynchronously updated.
- Client must get the *latest possible version* of the data [Lamport 1979, ABD 1995].



Distributed Key-value Stores

1. *Asynchrony*

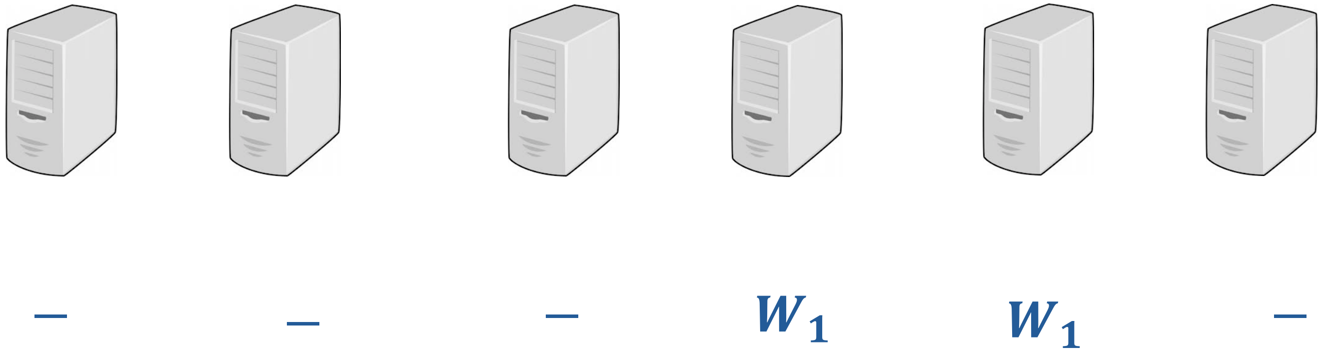
   Data updates may not arrive at all servers simultaneously.



Distributed Key-value Stores

1. *Asynchrony*

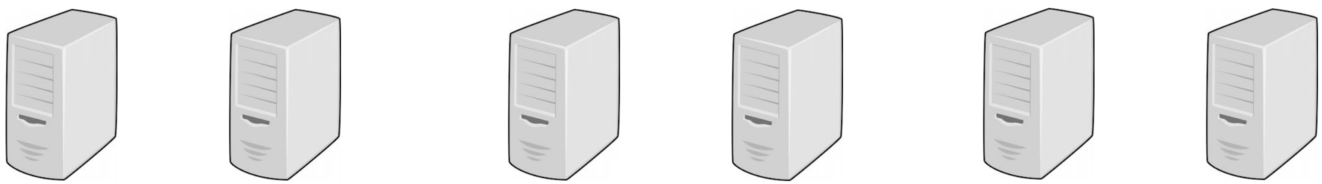
   Data updates may not arrive at all servers simultaneously.



Distributed Key-value Stores

1. *Asynchrony*

   Data updates may not arrive at all servers simultaneously.



–	–	–	W_1	W_1	–
W_2	–	W_2	W_2	W_2	–

Distributed Key-value Stores

1. *Asynchrony*

   Data updates may not arrive at all servers simultaneously.

2. *Decentralized Nature*

   A server may not be aware of which updates received by others.

$S(i) = \{2\}$

$S(j) = \{1, 2\}$

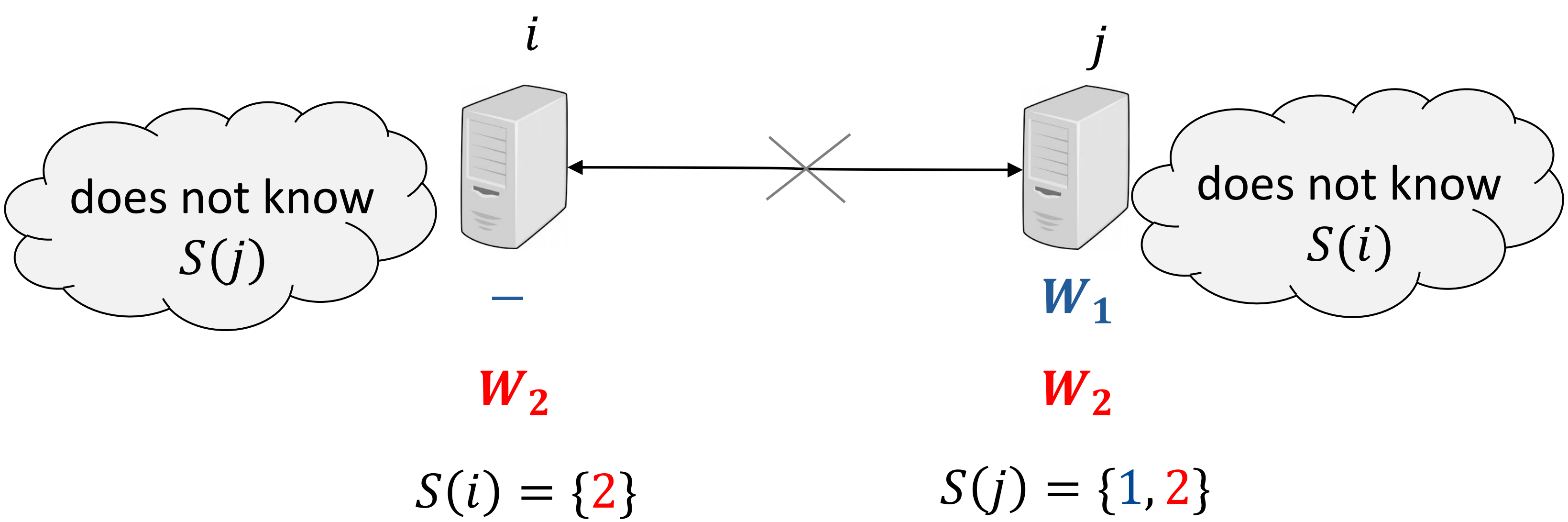
Distributed Key-value Stores

1. *Asynchrony*

   Data updates may not arrive at all servers simultaneously.

2. *Decentralized Nature*

   A server may not be aware of which updates received by others.



Distributed Key-value Stores

1. *Asynchrony*

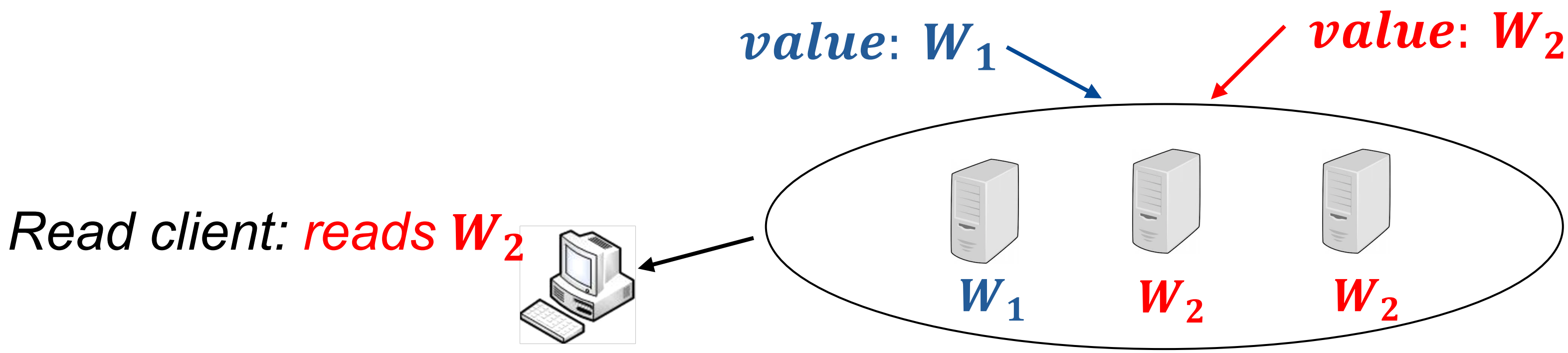
   Data updates may not arrive at all servers simultaneously.

2. *Decentralized Nature*

   A server may not be aware of which updates received by others.

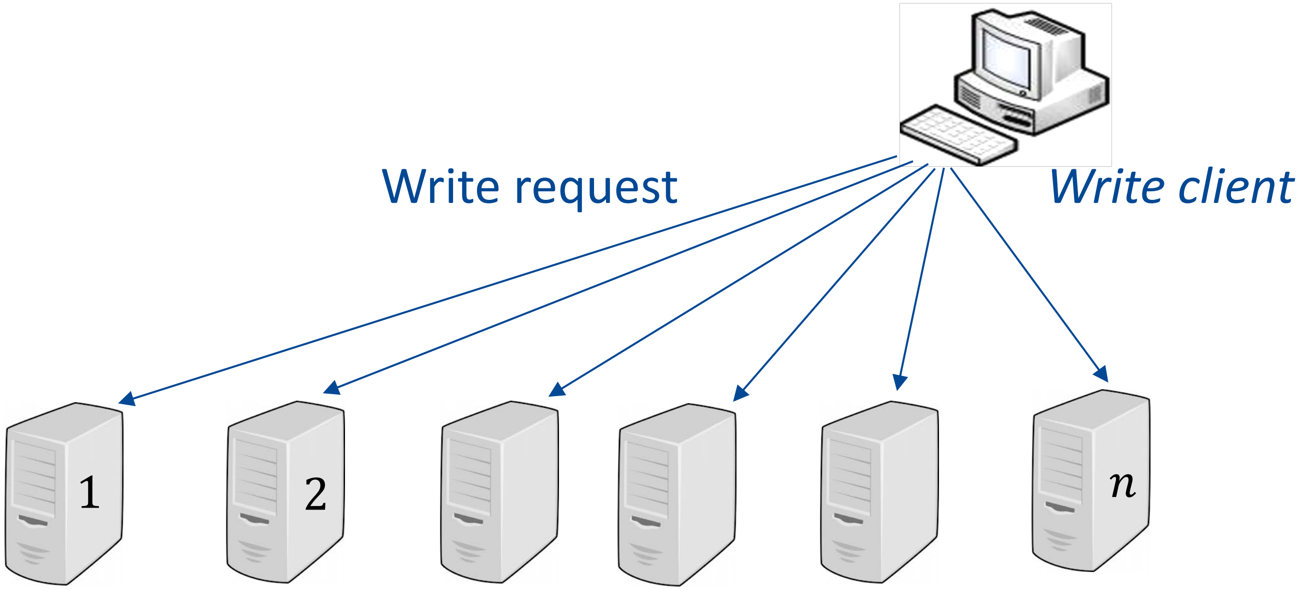
3. *Consistency*

   A client must retrieve the latest possible update.



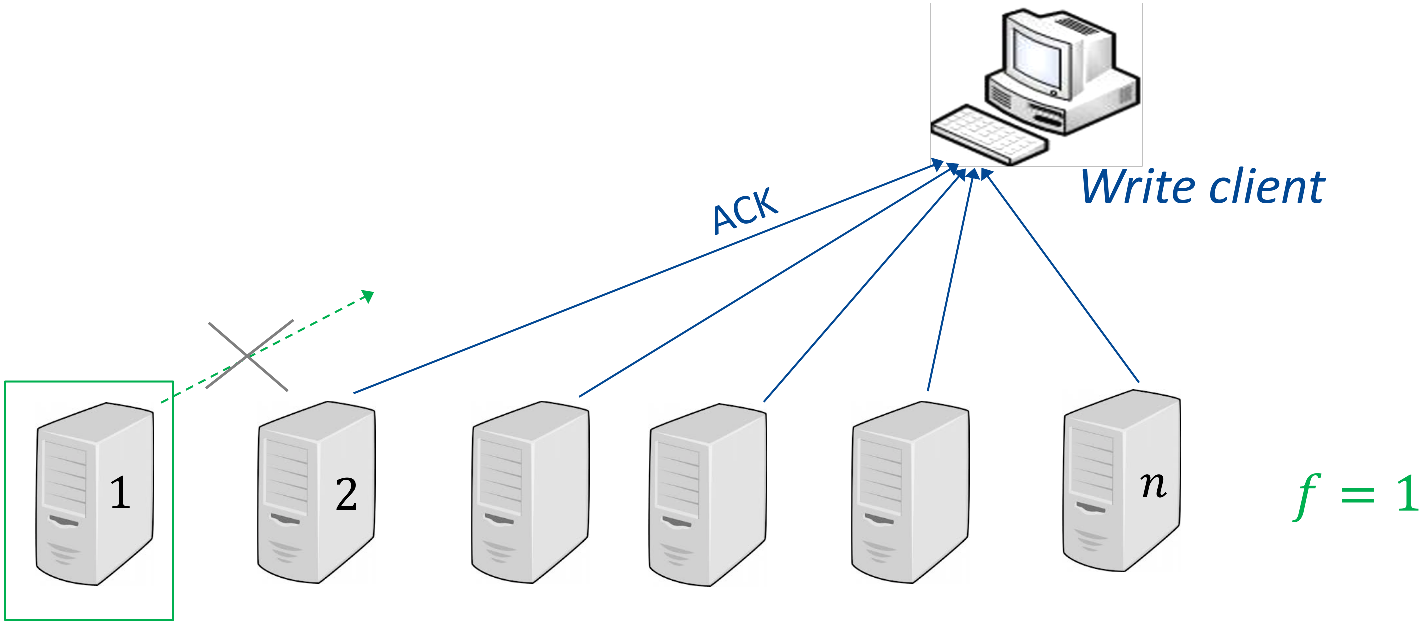
How to handle asynchrony & failures?

- Fault tolerance: f failures
- A complete write: write to $c_W \leq n - f$ servers



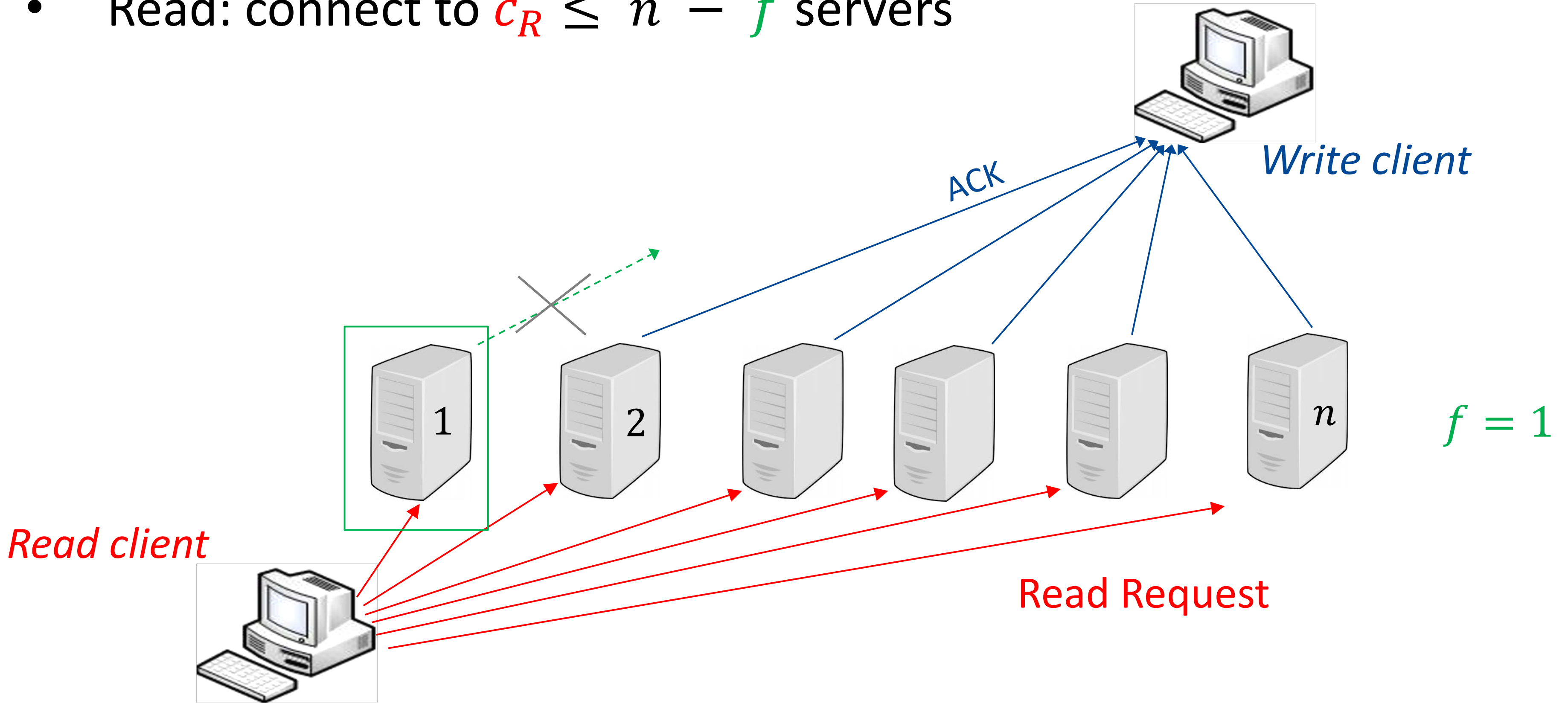
How to handle asynchrony & failures?

- Fault tolerance: f failures
- A complete write: write to $c_W \leq n - f$ servers



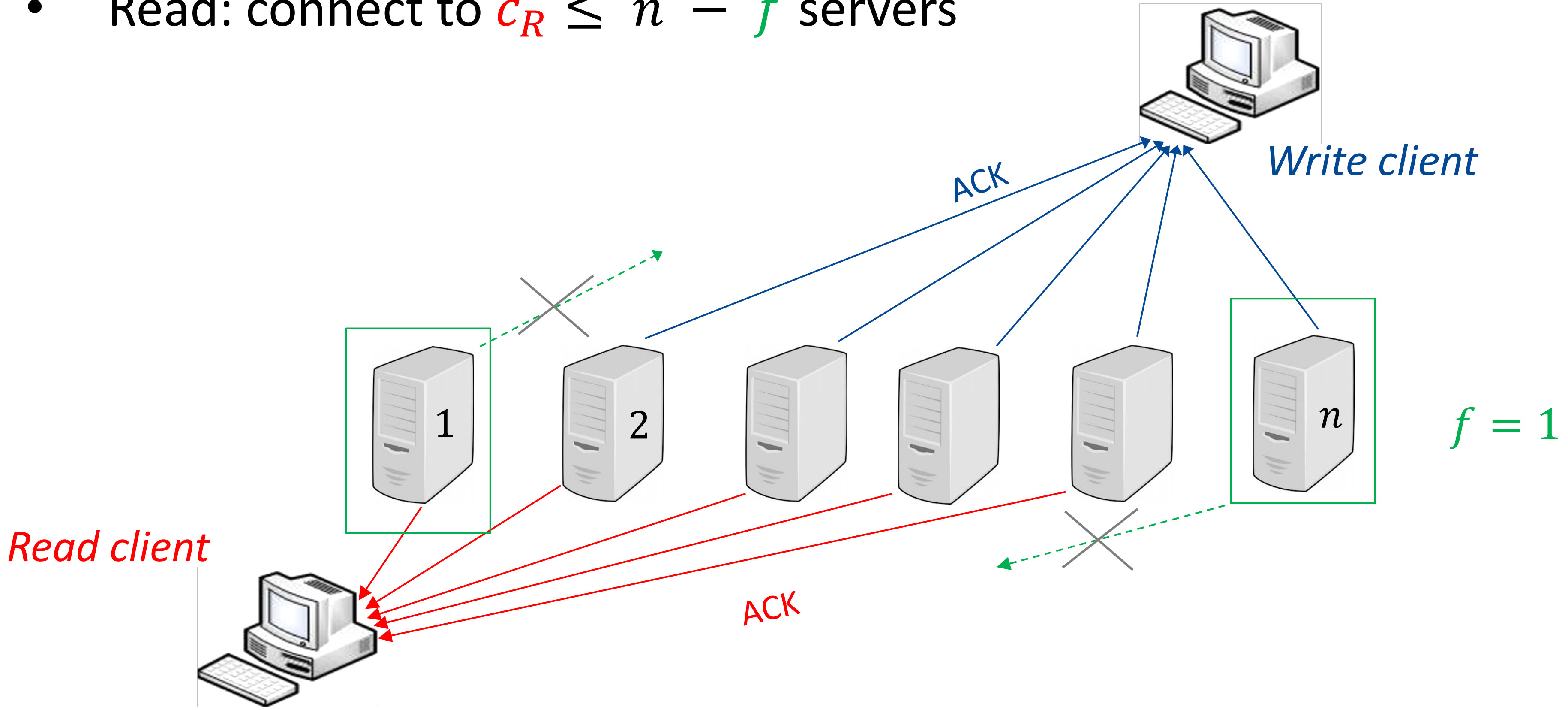
How to handle asynchrony & failures?

- Fault tolerance: f failures
- A complete write: write to $c_W \leq n - f$ servers
- Read: connect to $c_R \leq n - f$ servers



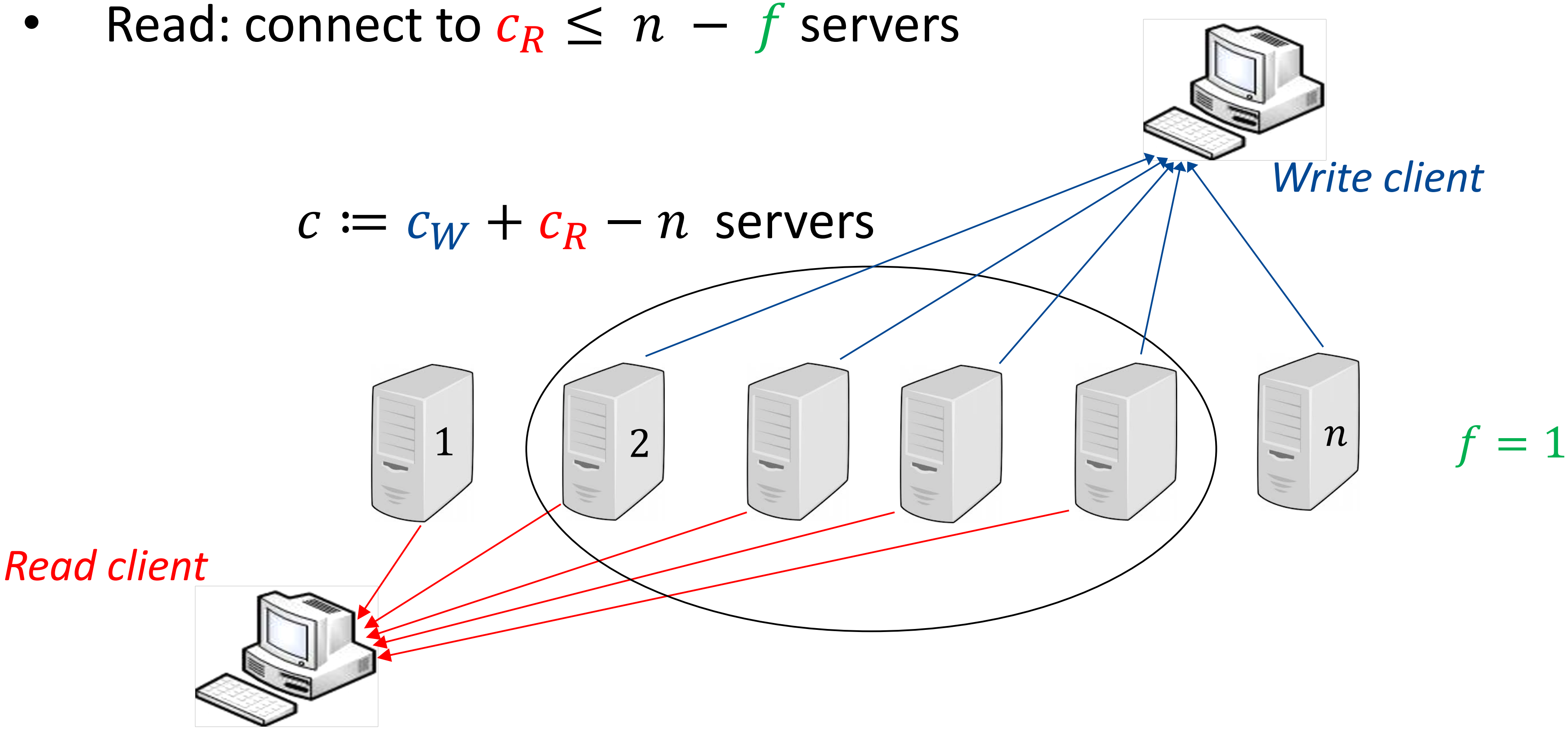
How to handle asynchrony & failures?

- Fault tolerance: f failures
- A complete write: write to $c_W \leq n - f$ servers
- Read: connect to $c_R \leq n - f$ servers



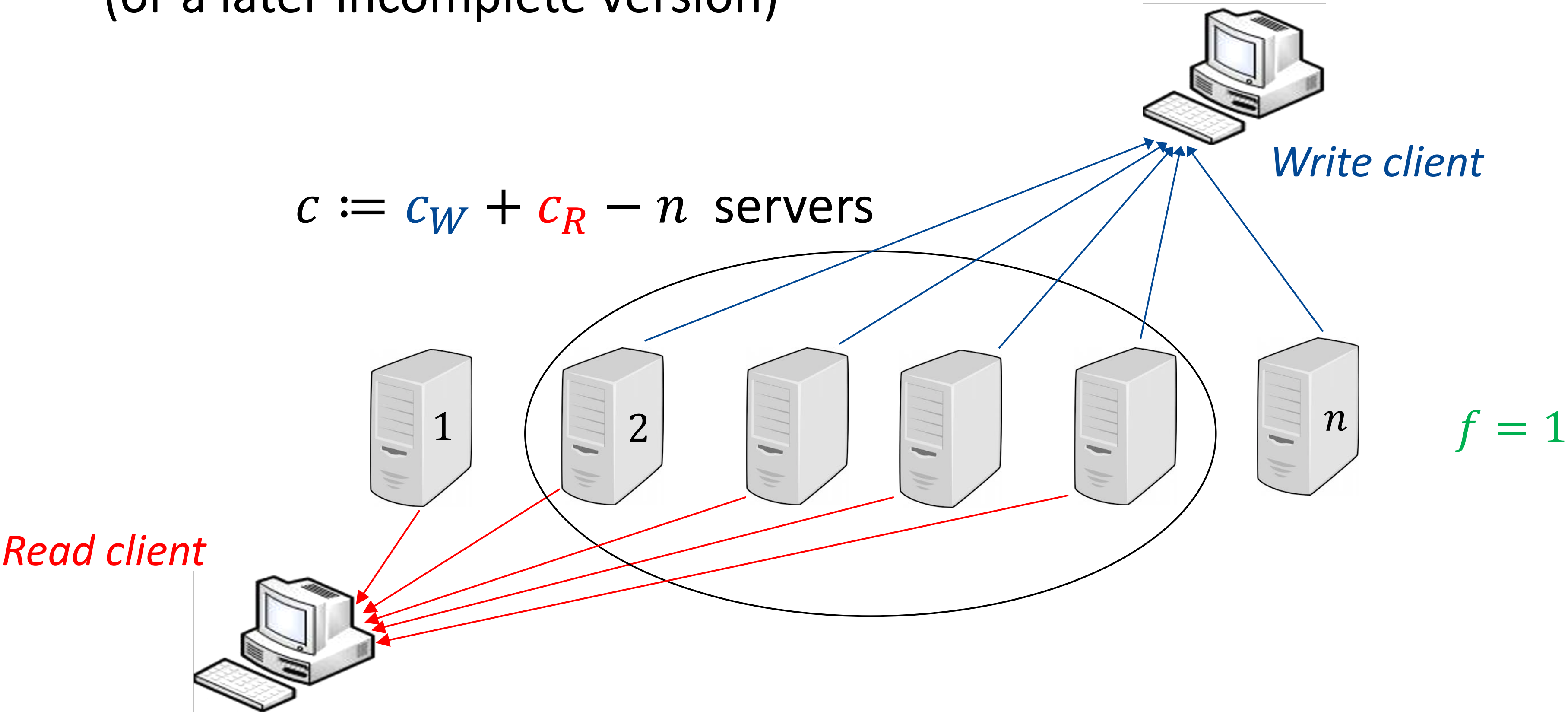
How to handle asynchrony & failures?

- Fault tolerance: f failures
- A complete write: write to $c_W \leq n - f$ servers
- Read: connect to $c_R \leq n - f$ servers



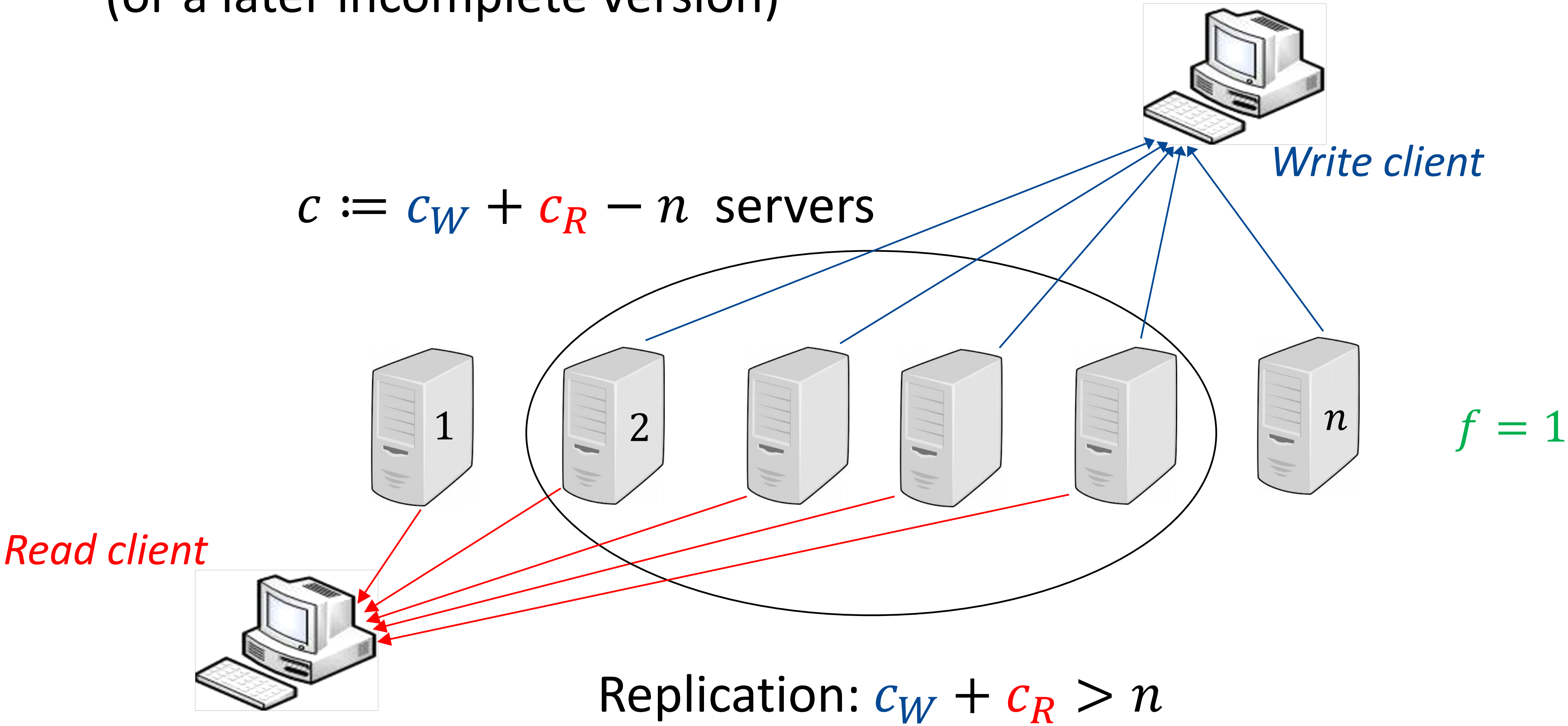
How to handle asynchrony & failures?

- **Strong Consistency:** decode the latest complete version (or a later incomplete version)



How to handle asynchrony & failures?

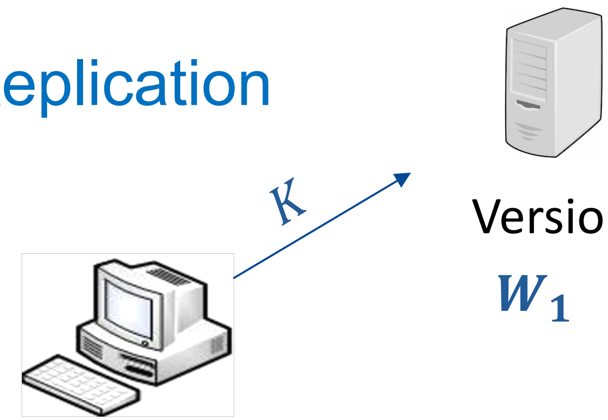
- **Strong Consistency:** decode the latest complete version (or a later incomplete version)



Replication: $c_W + c_R > n$

Background: Replication

Replication

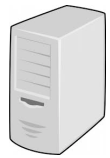


Write client

Version	no of bits
W_1	K

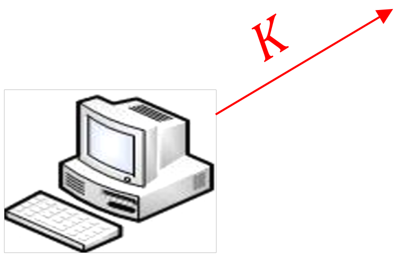
Background: Replication

Replication



Storage Cost$= K$

node stores only latest version

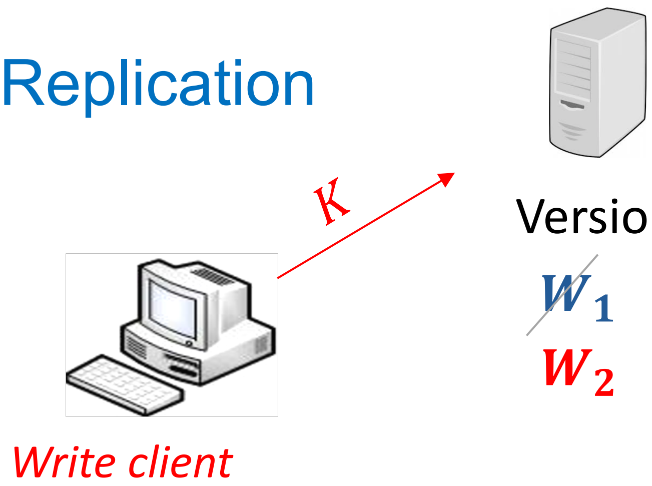


Write client

Version	no of bits
~~W_1~~	~~K~~
W_2	K

Background: Replication

Replication



Version	no of bits
~~W_1~~	~~K~~
W_2	K

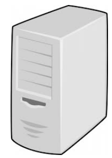
Storage Cost$= K$

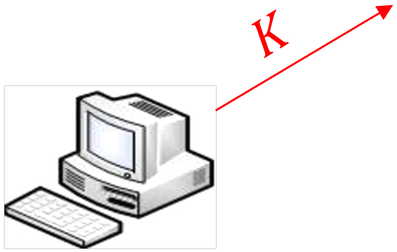
node stores only latest version

Significant Communication and Storage Costs

Background: Replication

Replication





Write client

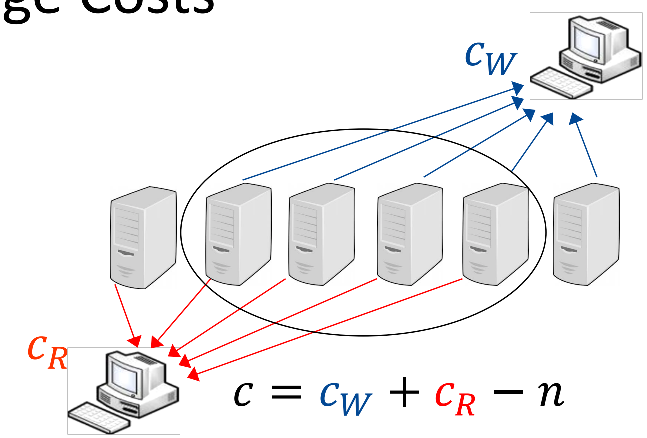
Version	no of bits
~~W_1~~	~~K~~
W_2	K

Storage Cost $= K$

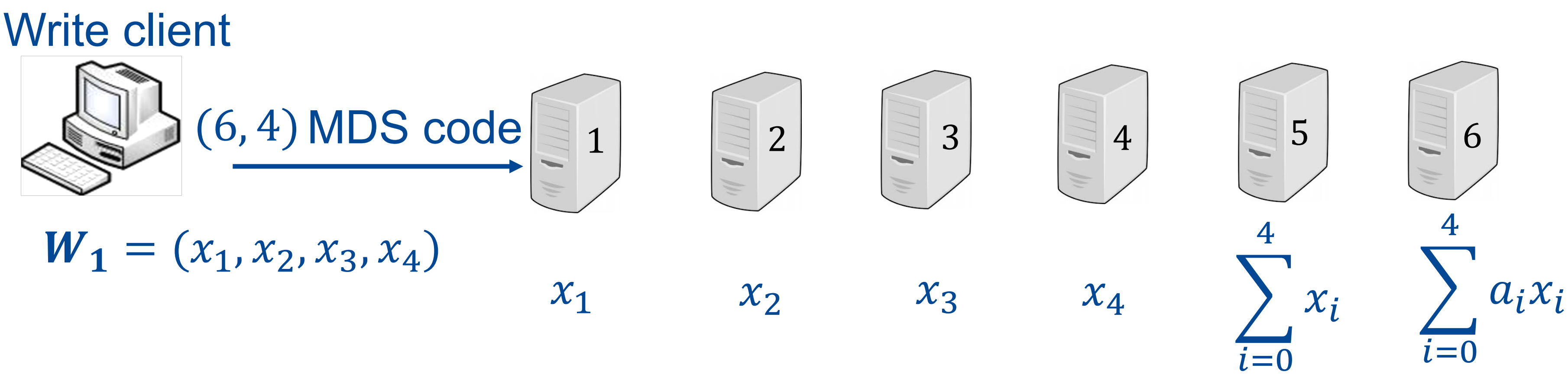
node stores only latest version

Significant Communication and Storage Costs

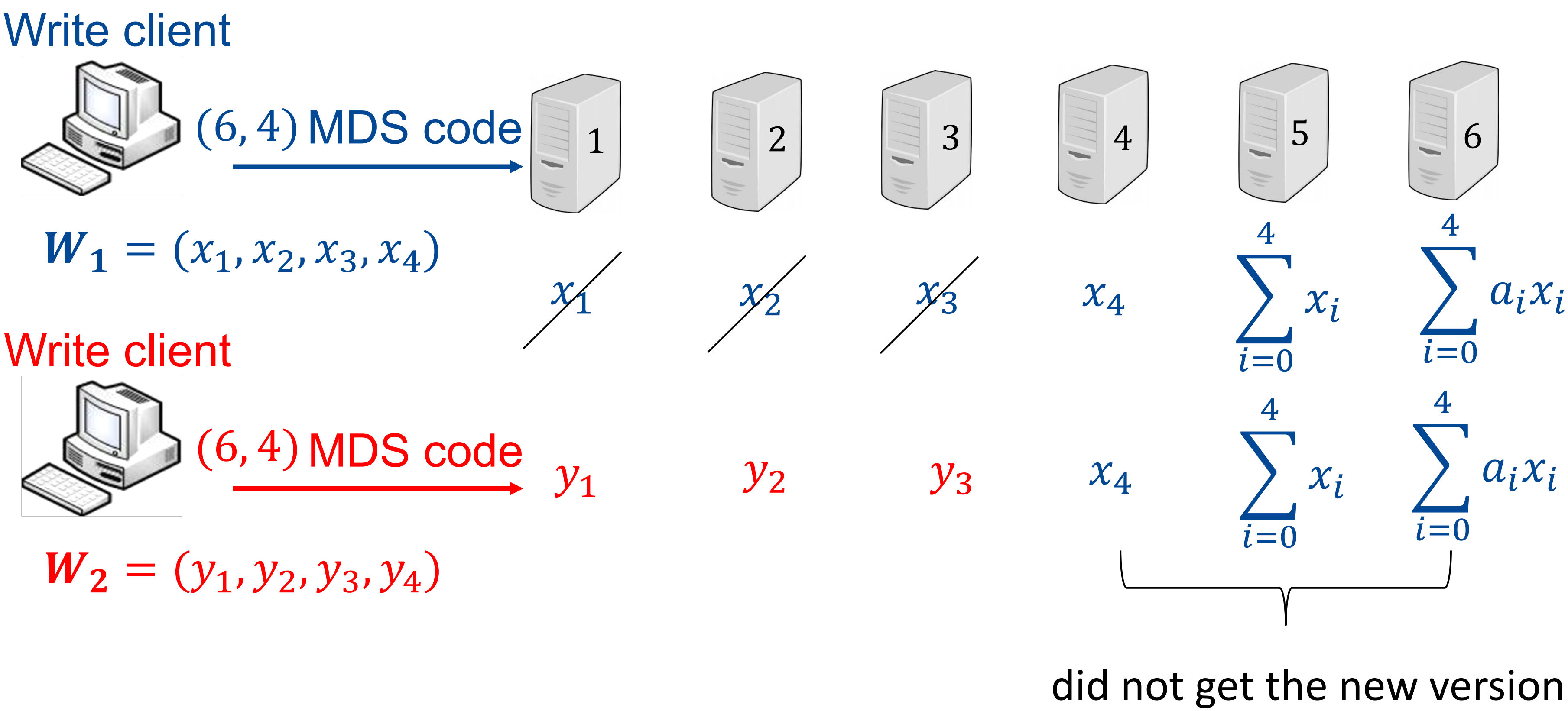
Use (n, c) MDS code, where each node stores $\frac{1}{c}$ of the data



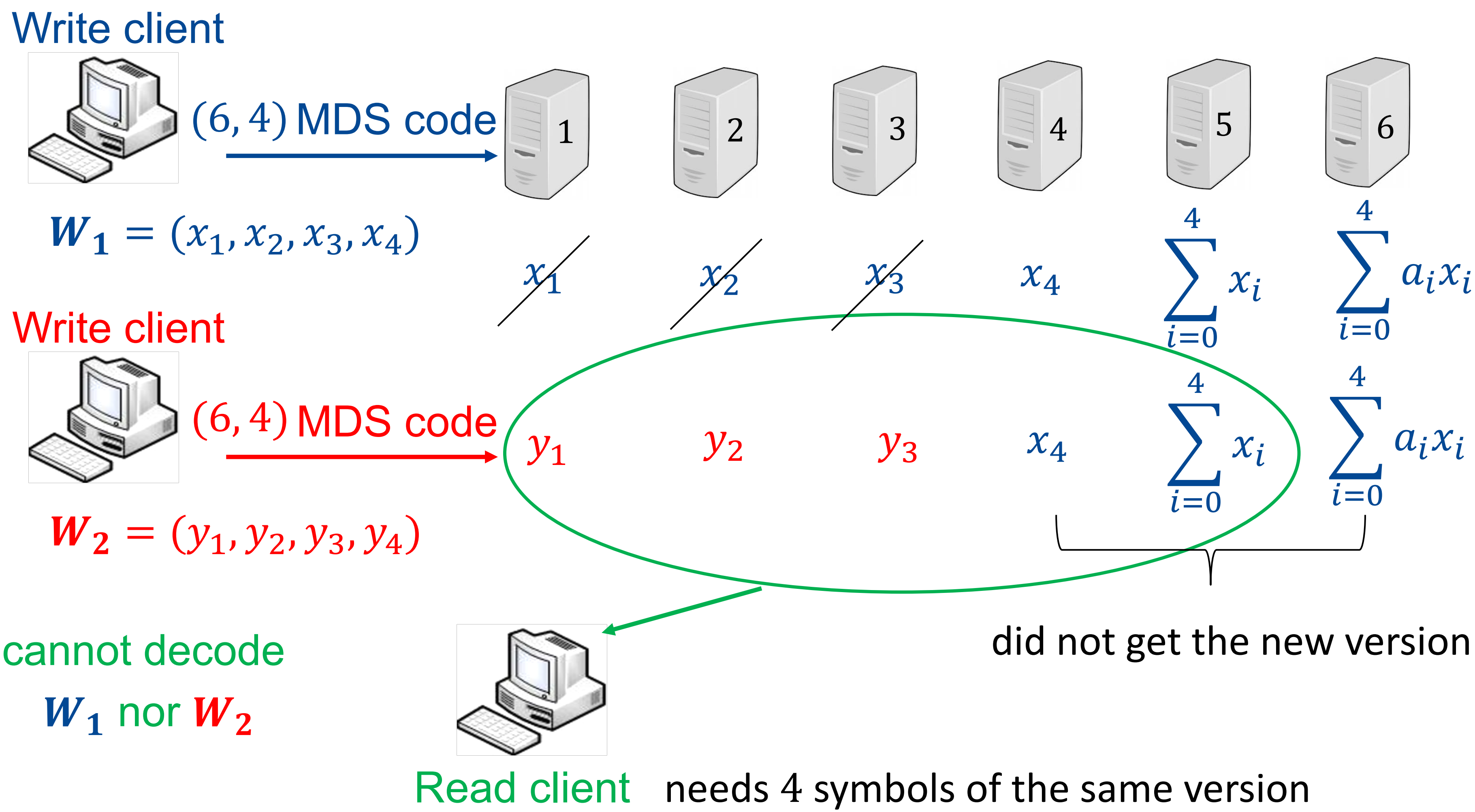
Background: Erasure Coding Challenges



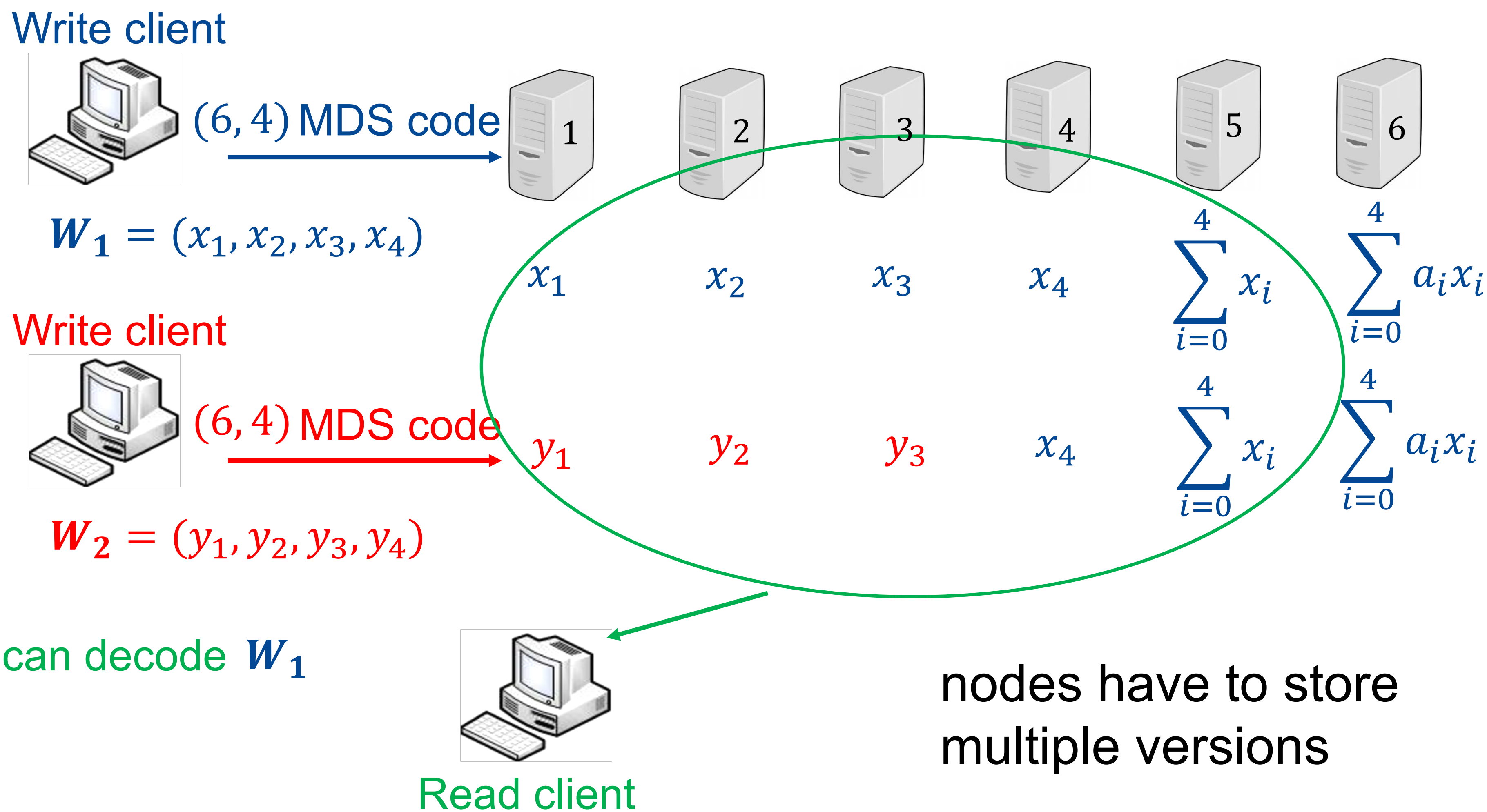
Background: Erasure Coding Challenges



Background: Erasure Coding Challenges



Background: Erasure Coding Challenges

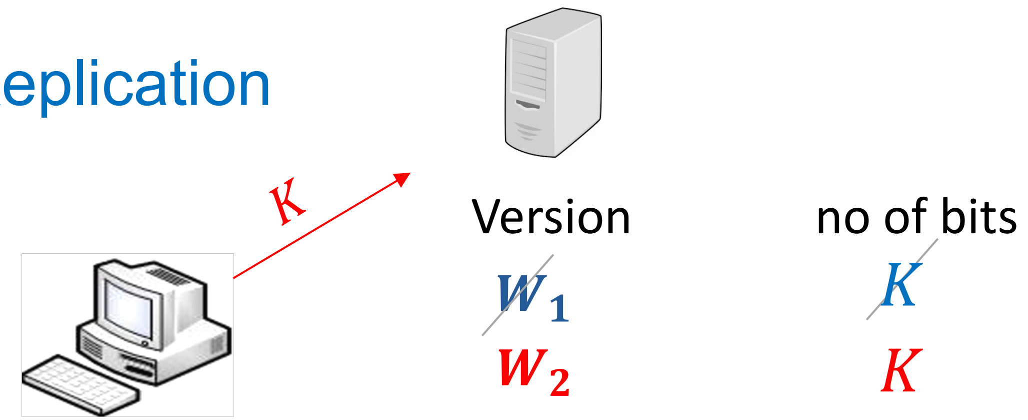


nodes have to store multiple versions

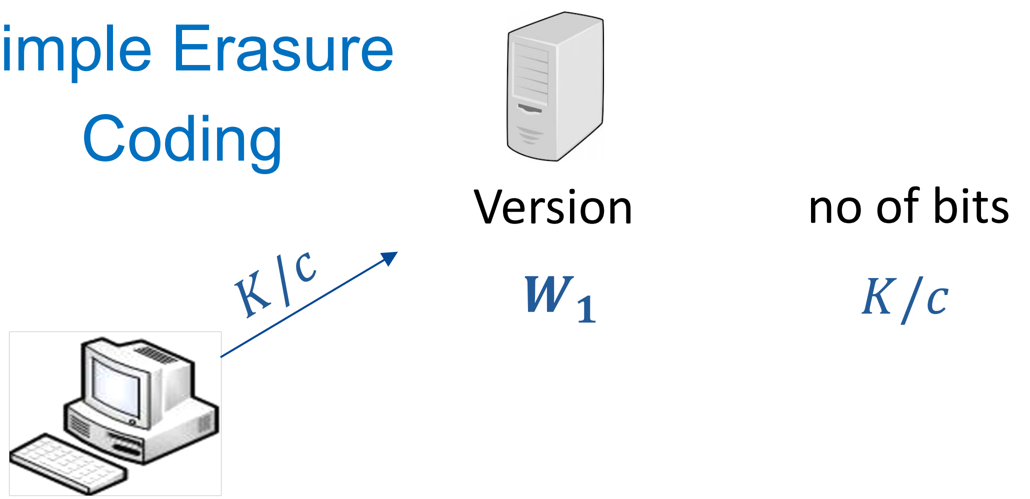
Background: Erasure Coding Challenges

Replication

Storage Cost$= K$

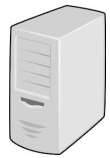


Simple Erasure Coding



Background: Erasure Coding Challenges

Replication



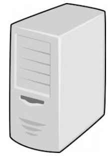
Storage Cost$= K$

K

Write client

Version	no of bits
~~W_1~~	~~K~~
W_2	K

Simple Erasure Coding



Storage Cost$= \nu \frac{1}{c} K$

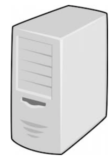
K/c

Write client

Version	no of bits
W_1	K/c
W_2	K/c

Background: Erasure Coding Challenges

Replication

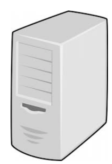


Version	no of bits
~~W_1~~	~~K~~
W_2	K

Storage Cost $= K$

node stores only latest version

Simple Erasure Coding



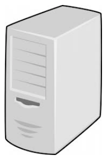
Version	no of bits
W_1	K/c
W_2	K/c

Storage Cost $= \nu \frac{1}{c} K$

node stores multiple versions

Background: Erasure Coding Challenges

Replication

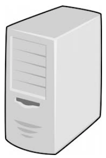


Storage Cost$= K$

Version	no of bits
~~W_1~~	~~K~~
W_2	K

Erasure coding gain

Simple Erasure Coding

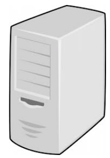


Storage Cost$= \nu \frac{1}{c} K$

Version	no of bits
W_1	K/c
W_2	K/c

Background: Erasure Coding Challenges

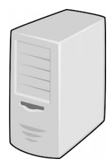
Replication



Storage Cost$= K$

Version	no of bits
~~W_1~~	~~K~~
W_2	K

Simple Erasure Coding



Version	no of bits
W_1	K/c
W_2	K/c

Erasure coding gain

Storage Cost$= \boxed{\nu} \frac{1}{c} K$

Offsets the gain

Background: Erasure Coding Challenges

Replication

Storage Cost $= K$

Version	no of bits
~~W_1~~	~~K~~
W_2	K

Simple Erasure Coding

Version	no of bits
W_1	K/c
W_2	K/c

Erasure coding gain

$$\text{Storage Cost} = \boxed{\nu}\,\frac{1}{c}\,K$$

Offsets the gain

Can we do better?

Background: Erasure Coding Challenges

Replication

Storage Cost$= K$

Version	no of bits
~~W_1~~	~~K~~
W_2	K

Simple Erasure Coding

Erasure coding gain

Storage Cost$= \boxed{\nu} \frac{1}{c} K$

Offsets the gain

Version	no of bits
W_1	K/c
W_2	K/c

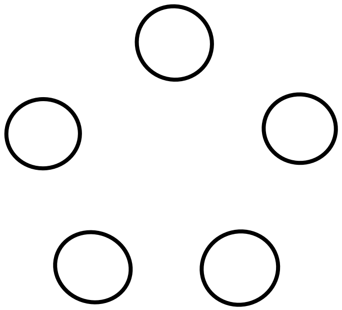
Can we do better?

[Wang et al. 2014] Storage Cost $\geq \boxed{\frac{\nu}{2}} \frac{1}{c} K - \Theta(1), \quad \nu < c$

Erasure-coded Key-value Stores with Side Information

Decentralized [Wang et al. 2014]

$$\text{Storage Cost} \geq \left(\frac{\boldsymbol{\nu}}{\boldsymbol{c}} - \frac{\nu(\nu-1)}{c^2} + o\left(\frac{1}{c^2}\right)\right)K$$

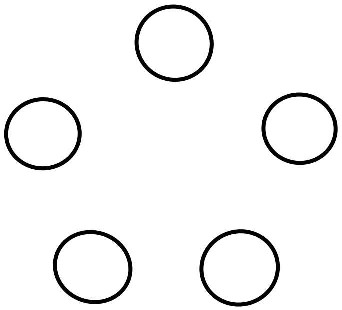


Erasure-coded Key-value Stores with Side Information

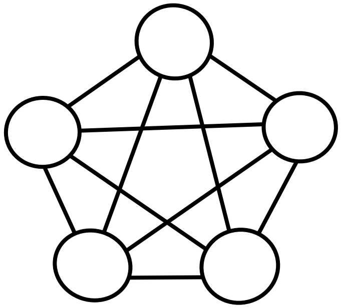
Decentralized [Wang et al. 2014]

Centralized

$$\text{Storage Cost} \geq \left(\frac{\boldsymbol{\nu}}{\boldsymbol{c}} - \frac{\nu(\nu-1)}{c^2} + o\left(\frac{1}{c^2}\right)\right)K$$



$$\text{Storage Cost} = \frac{\mathbf{1}}{\boldsymbol{c}}\,K$$

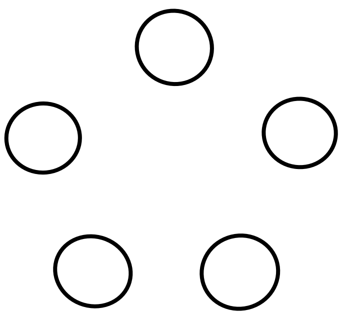


Erasure-coded Key-value Stores with Side Information

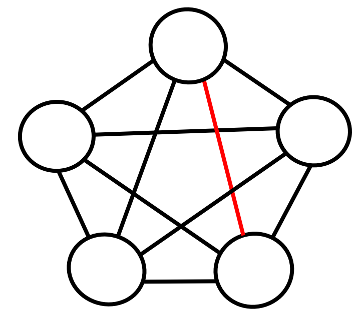
Decentralized [Wang et al. 2014]

Centralized

$$\text{Storage Cost} \geq \left(\frac{\boldsymbol{v}}{\boldsymbol{c}} - \frac{v(v-1)}{c^2} + o\left(\frac{1}{c^2}\right)\right)K$$



$$\text{Storage Cost} = \frac{\mathbf{1}}{\boldsymbol{c}}K$$



High Latency

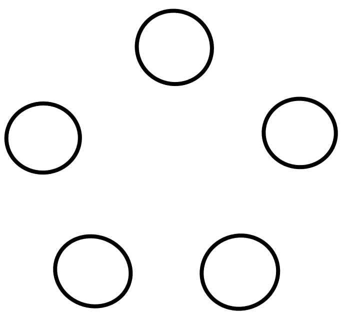
Geo-distributed key-value store

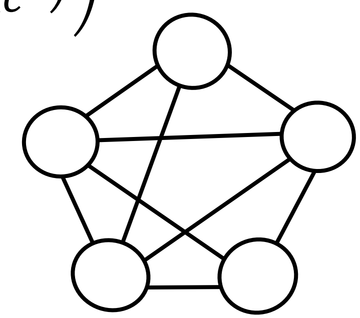
Erasure-coded Key-value Stores with Side Information

Decentralized [Wang et al. 2014]

Centralized

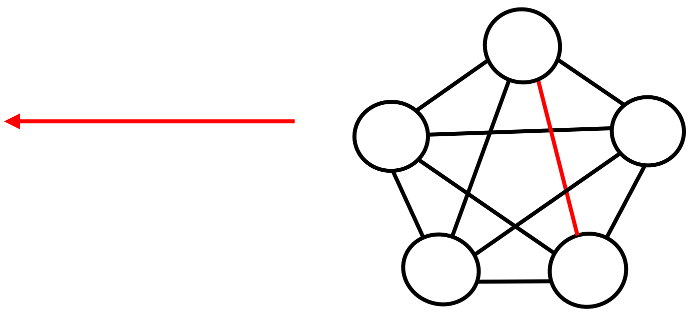
$$\text{Storage Cost} \geq \left(\frac{\boldsymbol{v}}{\boldsymbol{c}} - \frac{v(v-1)}{c^2} + o\left(\frac{1}{c^2}\right)\right)K$$





Storage Cost?

$$\text{Storage Cost} = \frac{\mathbf{1}}{\boldsymbol{c}}K$$



High Latency

Geo-distributed key-value store

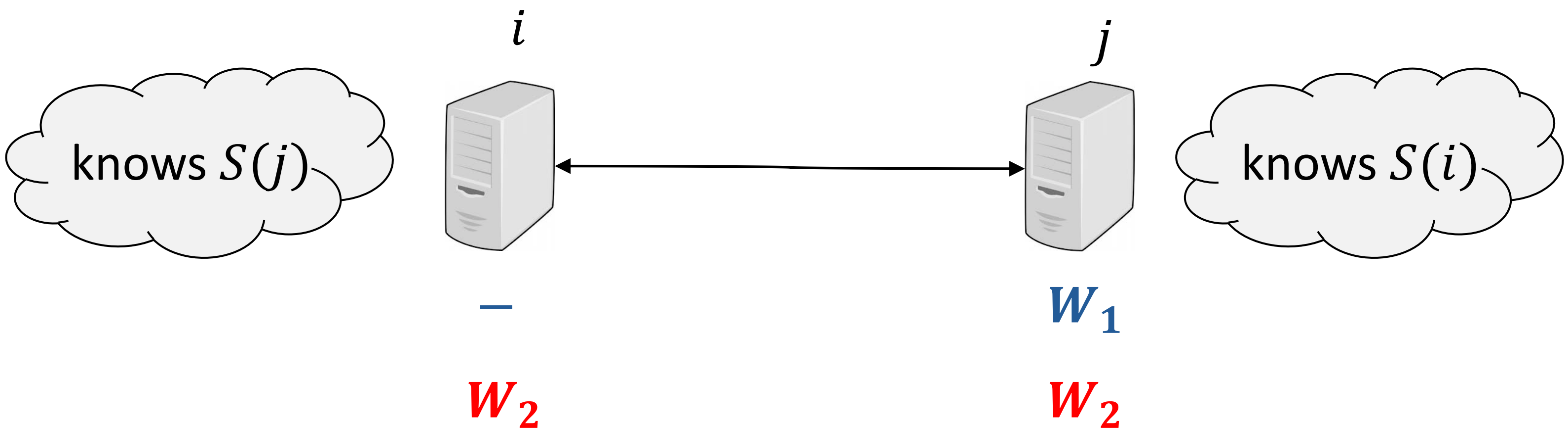
This Work: Coding with Partial Side Information

Latency-Storage Trade-off

Coding with Side Information

- Topology is given by a directed graph with degree H

$$\mathcal{G} = (\mathcal{N}, \mathcal{E})$$

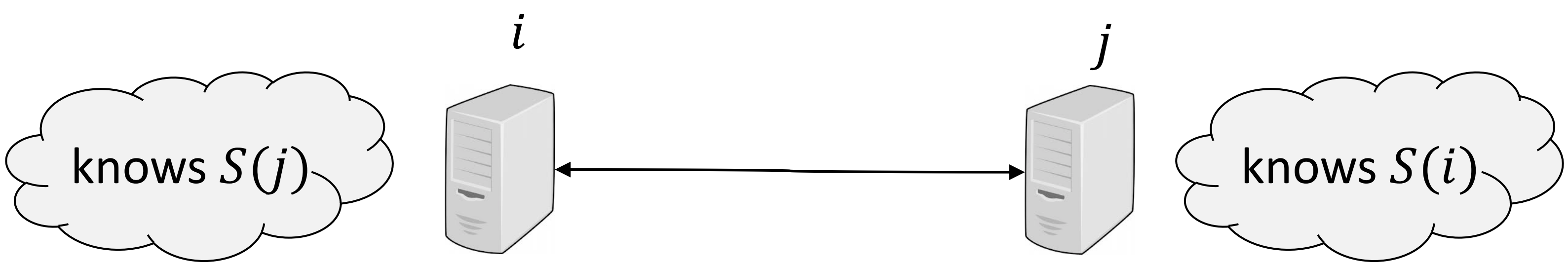


$S(i) = \{2\}$ $S(j) = \{1, 2\}$

Coding with Side Information

- Topology is given by a directed graph with degree H

$$\mathcal{G} = (\mathcal{N}, \mathcal{E})$$



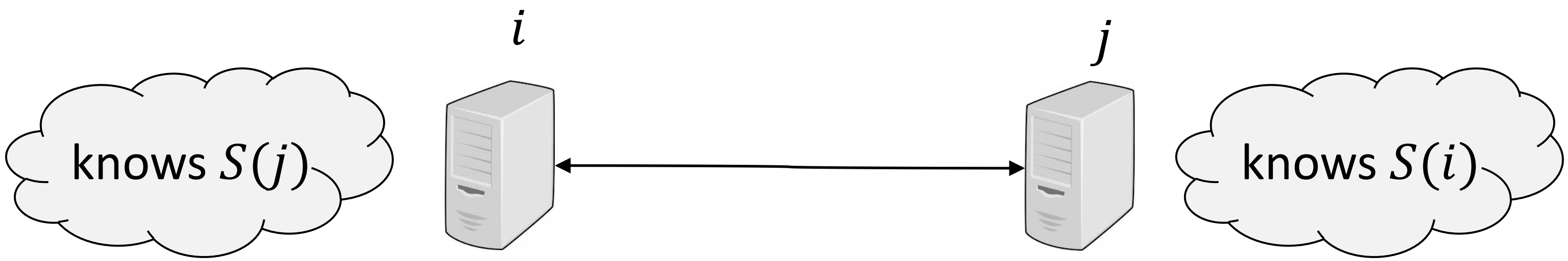
$-$	W_1
W_2	W_2
$S(i) = \{2\}$	$S(j) = \{1, 2\}$

Decoding Requirement: latest complete version (or a later version)

Coding with Side Information

- Topology is given by a directed graph with degree H

$$\mathcal{G} = (\mathcal{N}, \mathcal{E})$$



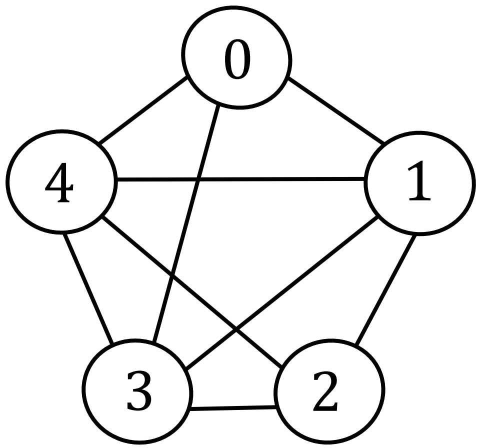
i	j
$-$	W_1
W_2	W_2
$S(i) = \{2\}$	$S(j) = \{1, 2\}$

Decoding Requirement: latest complete version (or a later version)

Idea: Can the servers guess which version is the latest complete?

Coding with Side Information: Challenges

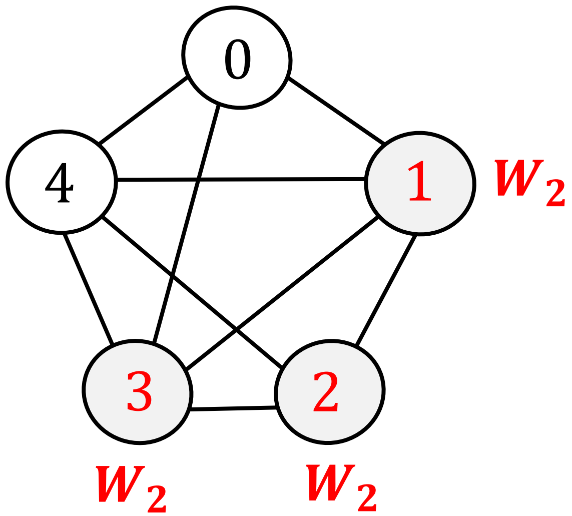
Can the servers guess which version is the latest complete?



$c_W = 4$

Coding with Side Information: Challenges

Can the servers guess which version is the latest complete?

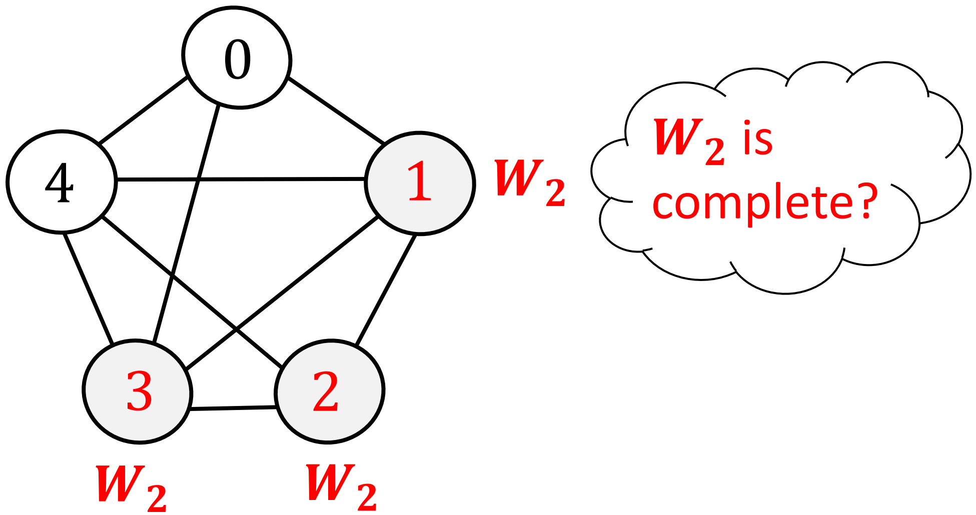


$c_W = 4$

W_2 is incomplete

Coding with Side Information: Challenges

Can the servers guess which version is the latest complete?

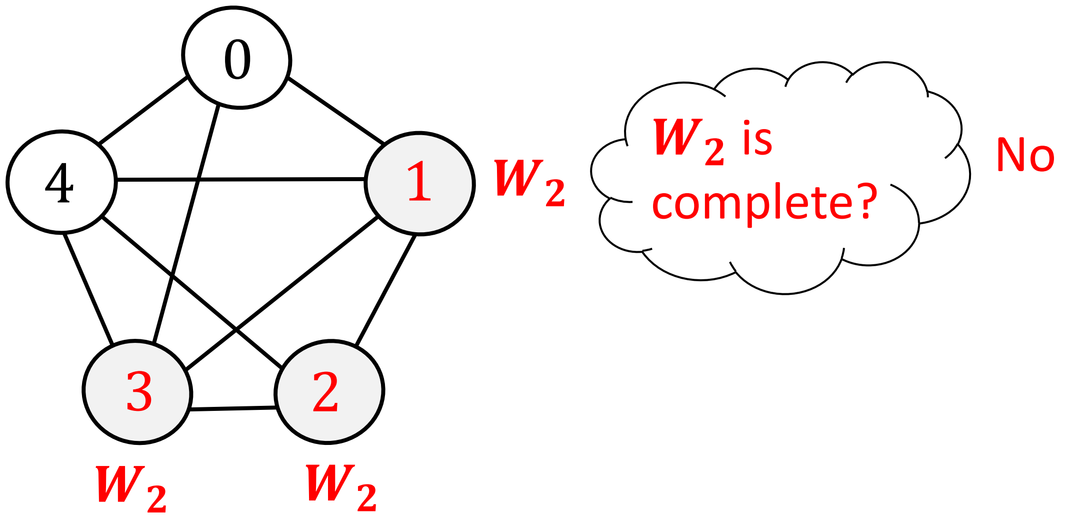


$c_W = 4$

W_2 is incomplete

Coding with Side Information: Challenges

Can the servers guess which version is the latest complete?

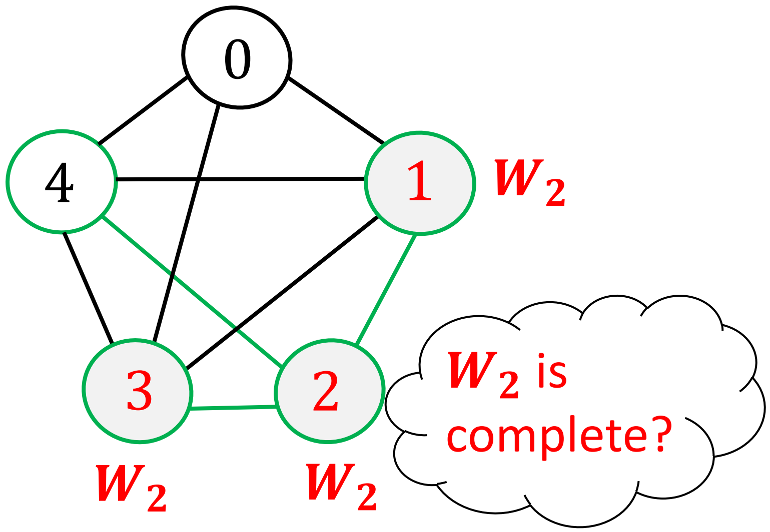


$c_W = 4$

W_2 is incomplete

Coding with Side Information: Challenges

Can the servers guess which version is the latest complete?

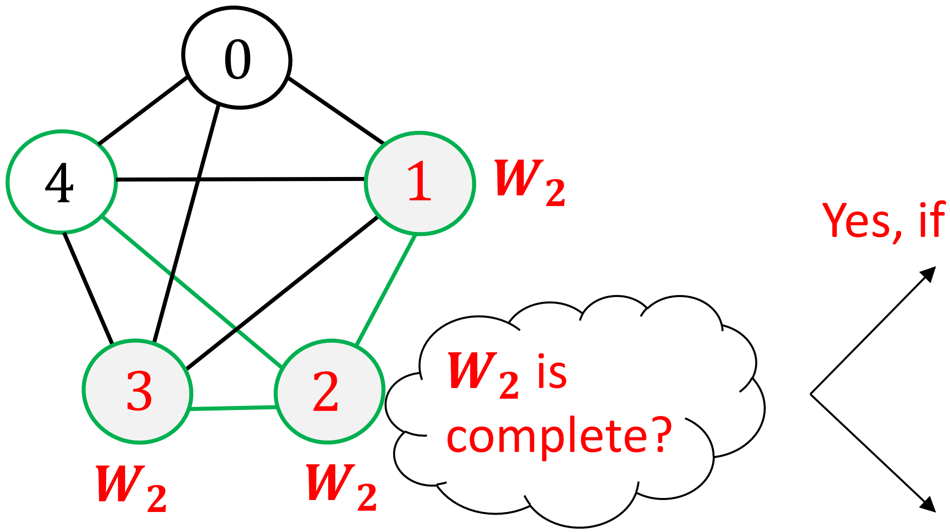


$c_W = 4$

W_2 is incomplete

Coding with Side Information: Challenges

Can the servers guess which version is the latest complete?



$c_W = 4$

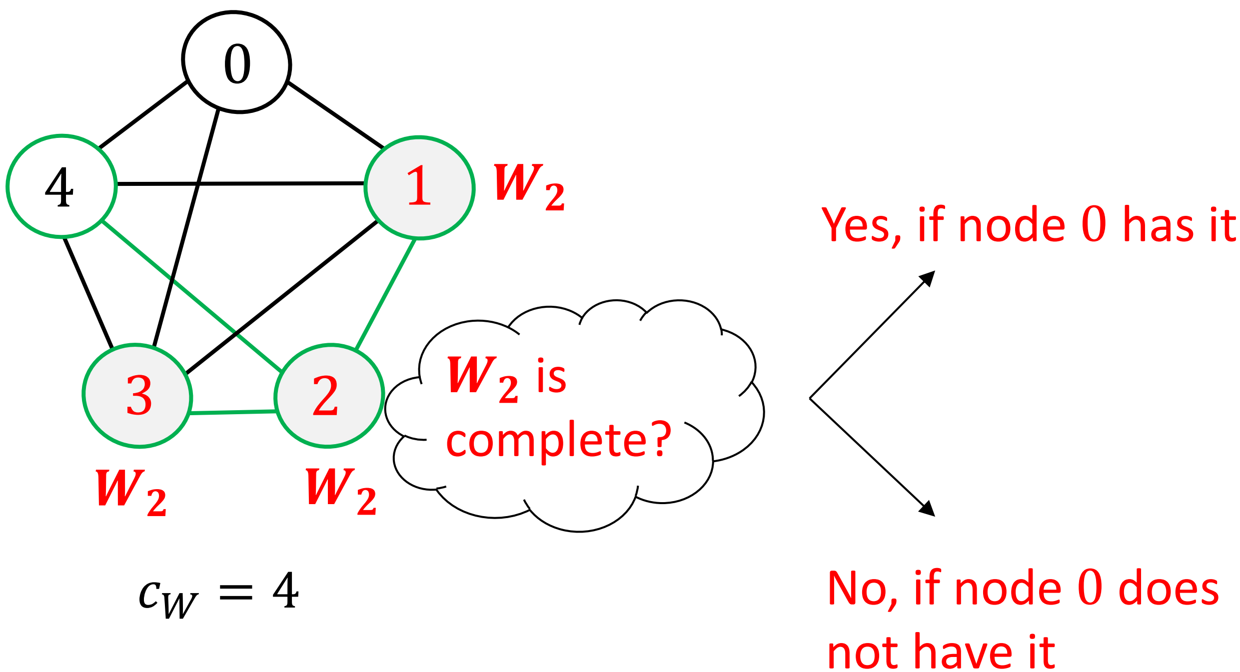
W_2 is incomplete

Yes, if node 0 has it

No, if node 0 does not have it

Coding with Side Information: Challenges

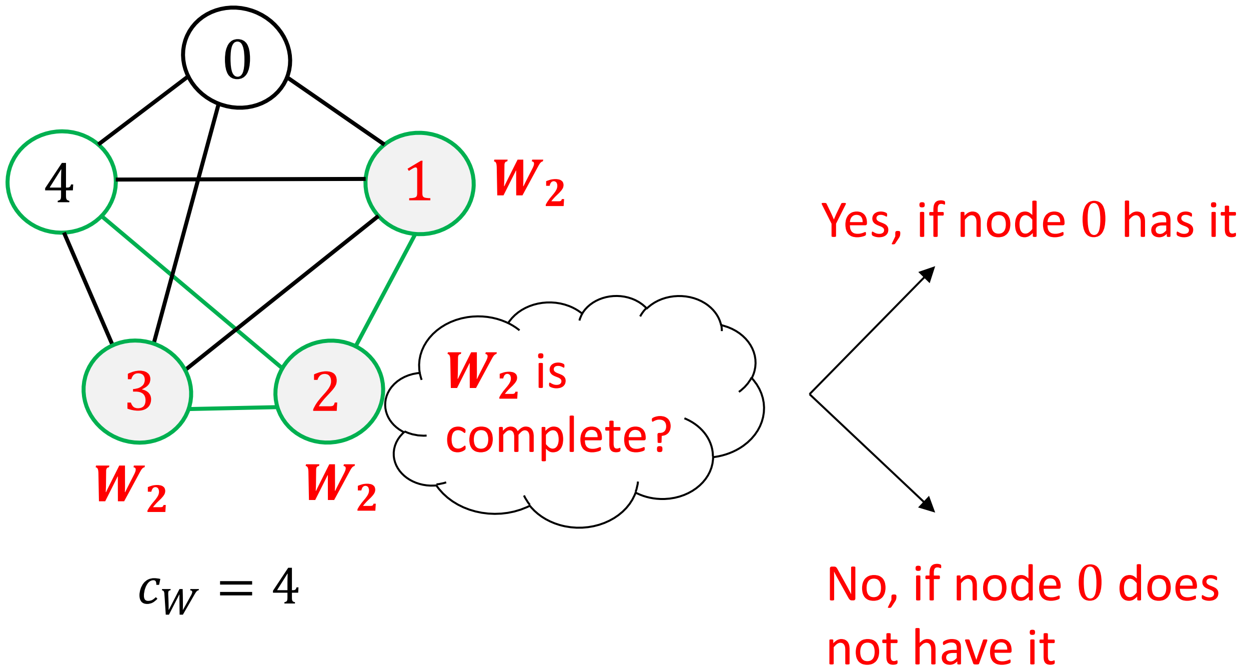
Can the servers guess which version is the latest complete?



node 2 does not know that W_2 is incomplete!

Coding with Side Information: Challenges

Can the servers guess which version is the latest complete?

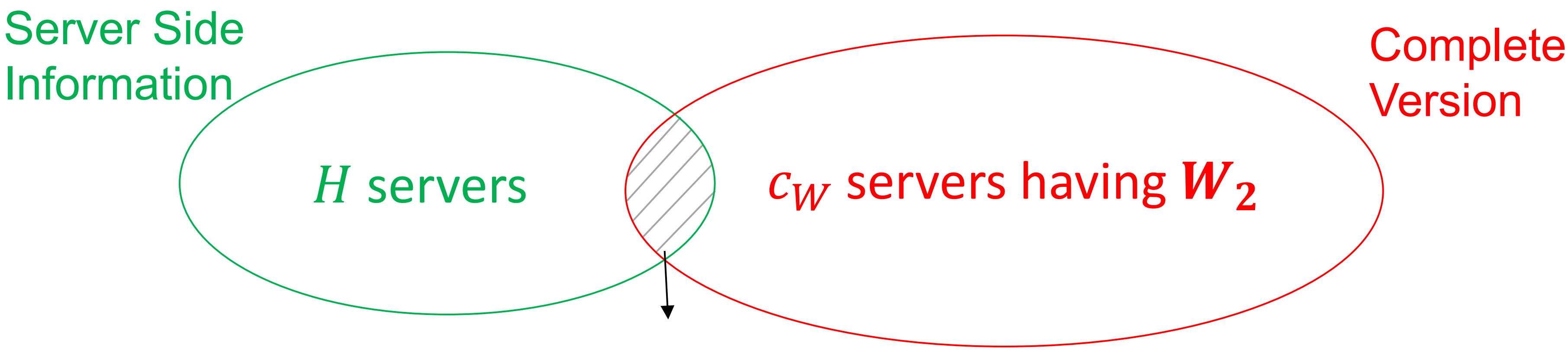


$c_W = 4$

node 2 does not know that W_2 is incomplete!

Given $\mathcal{G}$, how many servers cannot guess correctly?

Coding with Side Information: Challenges

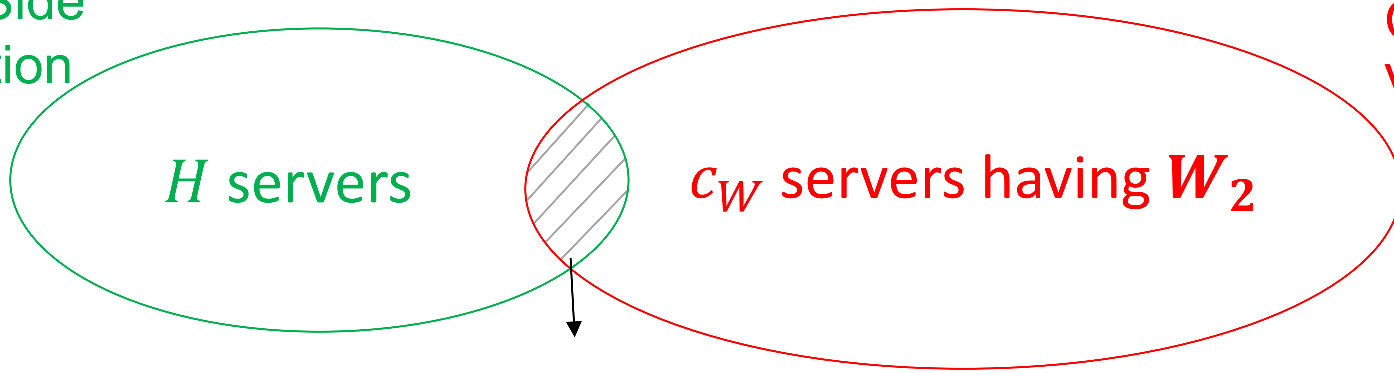


observes at least $c_W + H - n$ servers having $\boldsymbol{W_2}$

Coding with Side Information: Challenges

Server Side Information

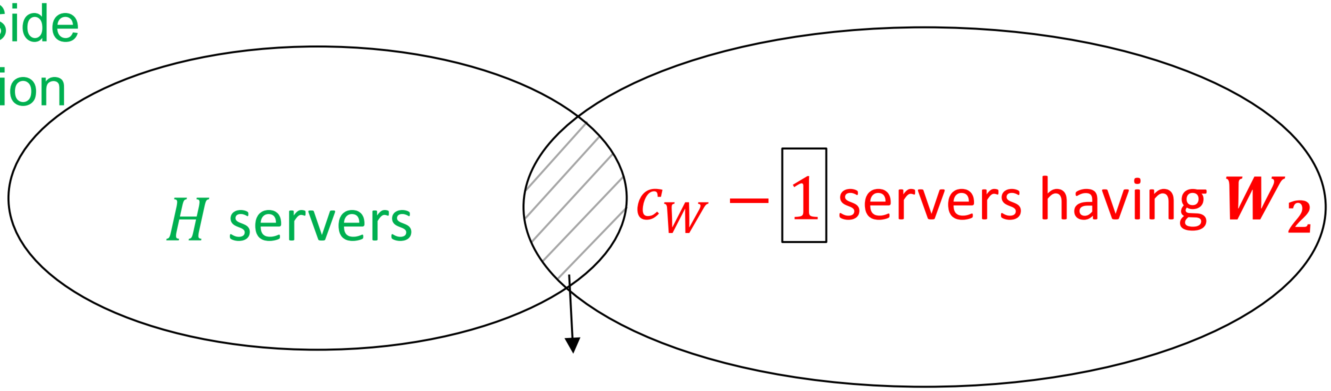
Complete Version



observes at least $c_W + H - n$ servers having $\boldsymbol{W_2}$

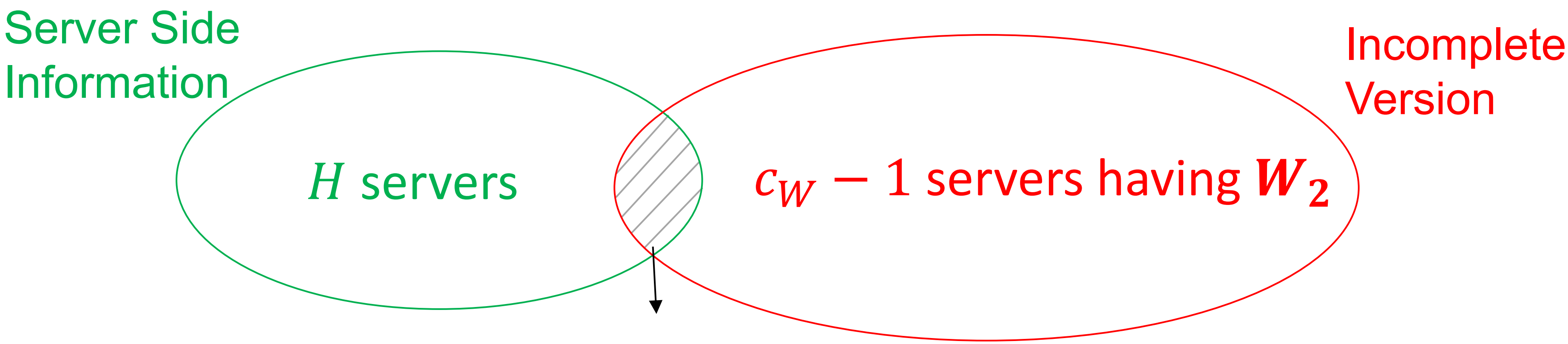
Server Side Information

Incomplete Version



may still observe at least $c_W + H - n$ servers having $\boldsymbol{W_2}$

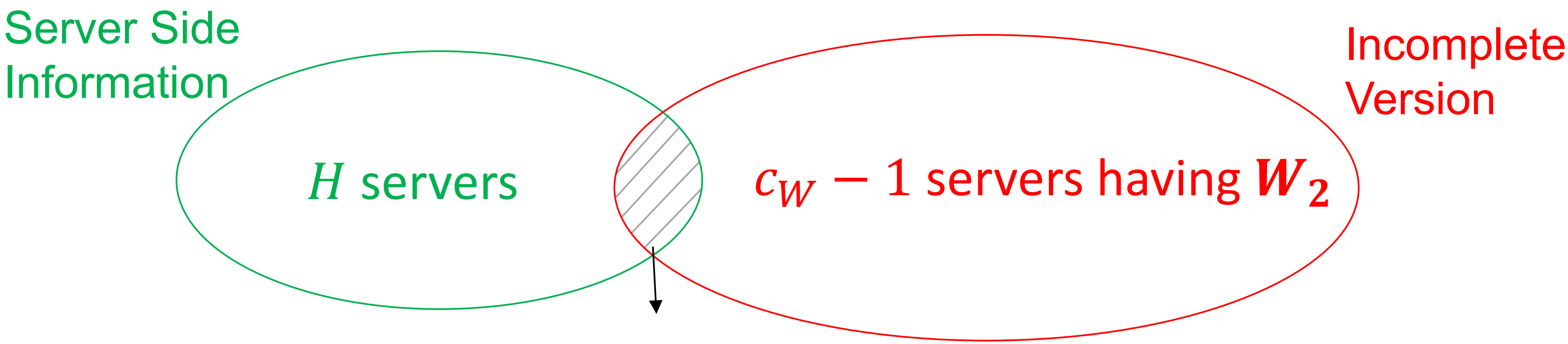
Coding with Side Information: Challenges



may still at least $c_W + H - n$ servers having $\boldsymbol{W_2}$

Given an incomplete version, how many servers may assume it is complete?

Coding with Side Information: Challenges

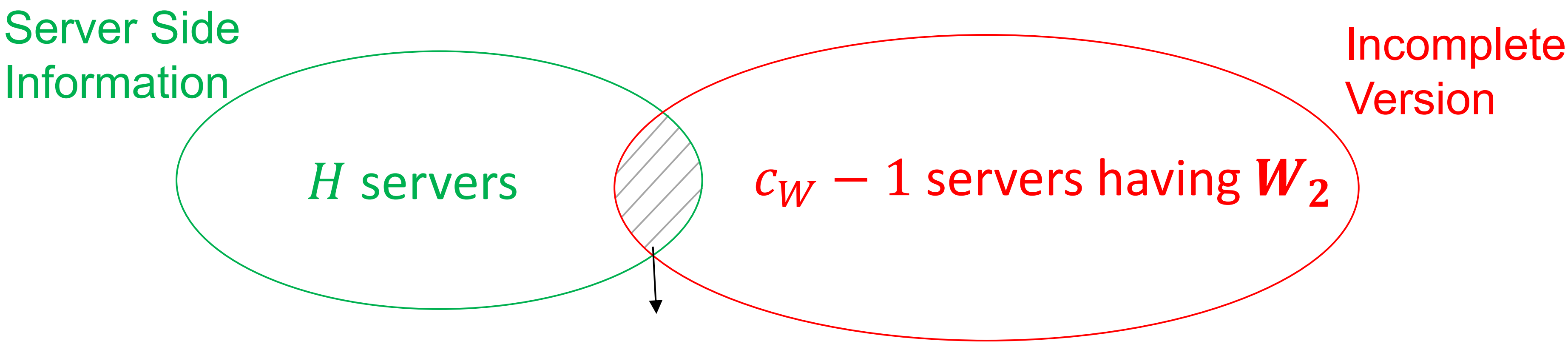


may still at least $c_W + H - n$ servers having $\boldsymbol{W_2}$

Given an incomplete version, how many servers may assume it is complete?

$$\bar{m}(\mathcal{G}) = \max_{\mathcal{G}'=(\mathcal{N}',\mathcal{E}')\subset\mathcal{G}:|\mathcal{N}'|=c_W-1} \left|\{i' \in \mathcal{N}': \deg^{+}_{\mathcal{G}'}(i') \geq c_W + H - n\}\right|$$

Coding with Side Information: Challenges



may still at least $c_W + H - n$ servers having $\boldsymbol{W_2}$

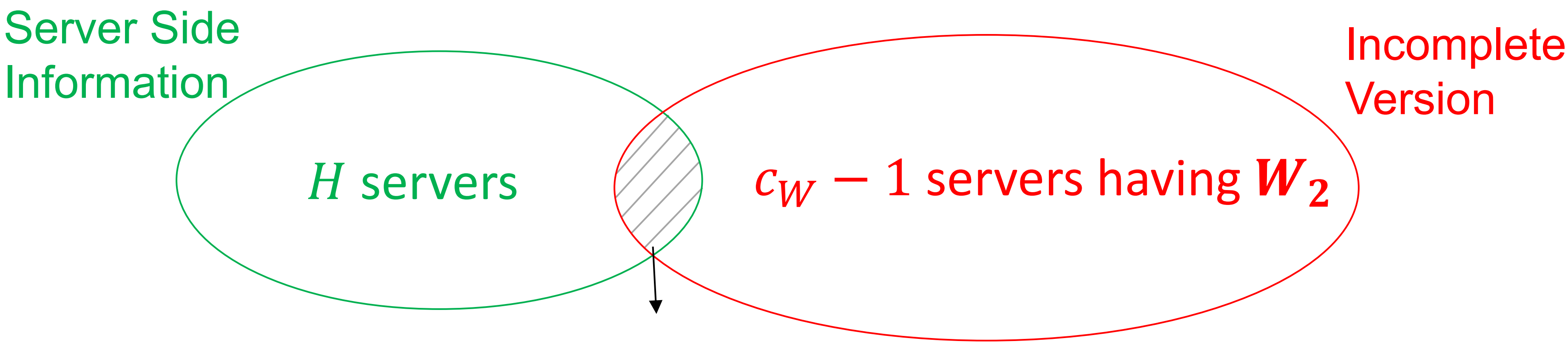
Given an incomplete version, how many servers may assume it is complete?

$$\bar{m}(\mathcal{G}) = \max_{\mathcal{G}'=(\mathcal{N}',\mathcal{E}')\subset\mathcal{G}:|\mathcal{N}'|=c_W-1} \left|\{i'\in \mathcal{N}': \deg^+_{\mathcal{G}'}(i') \geq c_W + H - n\}\right|$$

We need to consider $\binom{n}{c_W - 1}$ graphs

Computationally challenging for large graphs

Coding with Side Information: Challenges



may still at least $c_W + H - n$ servers having $\boldsymbol{W_2}$

Given an incomplete version, how many servers may assume it is complete?

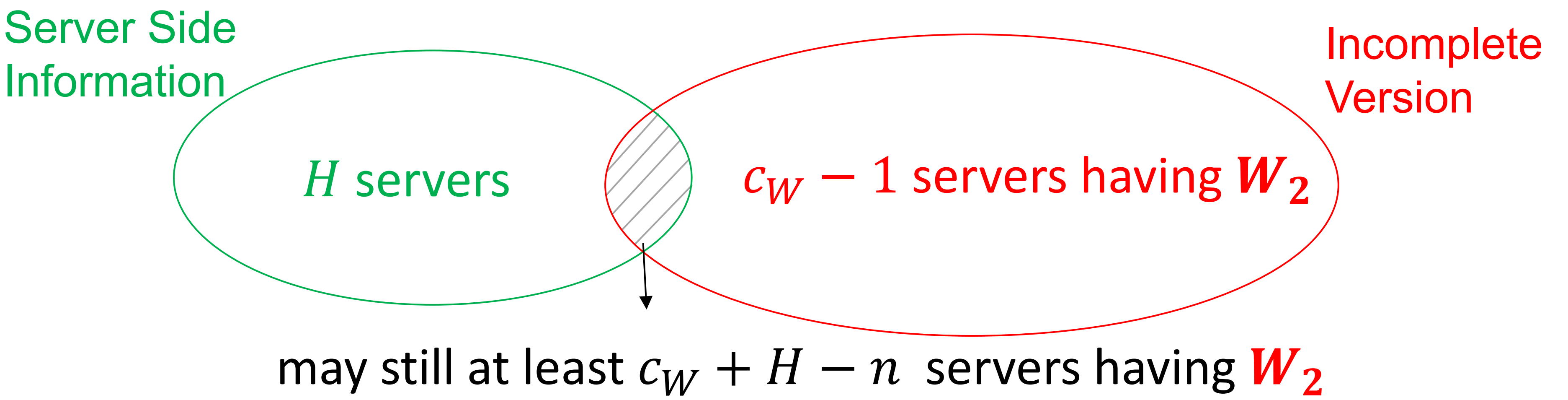
$$\bar{m}(\mathcal{G}) = \max_{\mathcal{G}'=(\mathcal{N}',\mathcal{E}')\subset\mathcal{G}:|\mathcal{N}'|=c_W-1} \left|\{i' \in \mathcal{N}' : \deg^{+}_{\mathcal{G}'}(i') \geq c_W + H - n\}\right|$$

We need to consider $\binom{n}{c_W - 1}$ graphs

Computationally challenging for large graphs

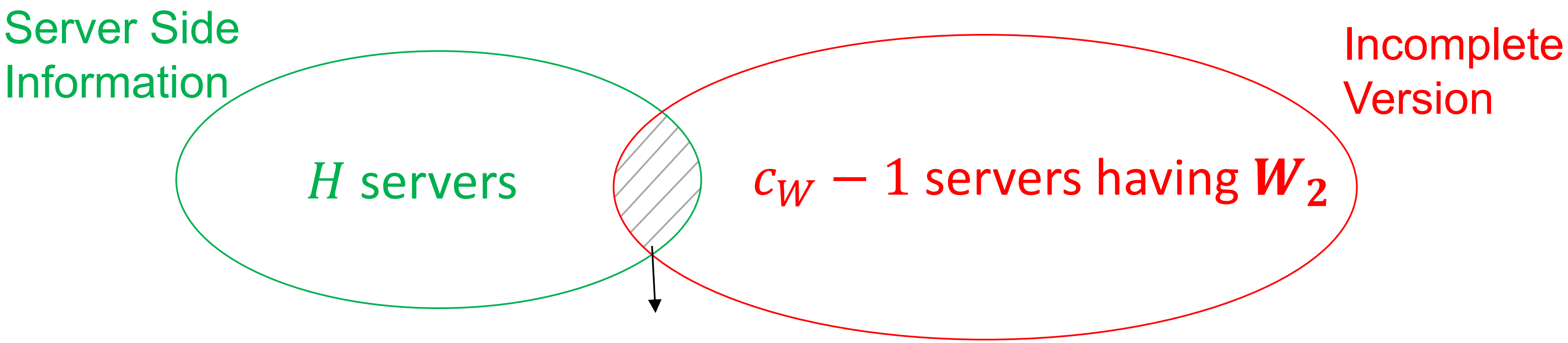
$$\bar{m}(\mathcal{G}) \leq (n - c_W + 1)\,(n - H)$$

Coding with Side Information: Construction



Coding Strategy: a server stores part of $\boldsymbol{W_2}$ if it observes at least $c_W + H - n$ servers having it.

Coding with Side Information: Construction



may still at least $c_W + H - n$ servers having $\boldsymbol{W_2}$

Coding Strategy: a server stores part of $\boldsymbol{W_2}$ if it observes at least $c_W + H - n$ servers having it.

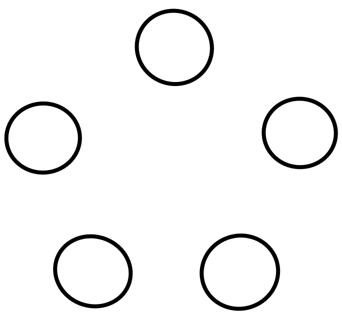
At most $\overline{m}(\mathcal{G})$ servers store $\boldsymbol{W_2}$ when it is incomplete.

$$\text{Storage Cost} = \left(\frac{\mathbf{1}}{\boldsymbol{c}} + \frac{(\nu - 1)\overline{m}(\mathcal{G})}{c^2} + o\left(\frac{\overline{m}(\mathcal{G})}{c^2}\right)\right)K$$

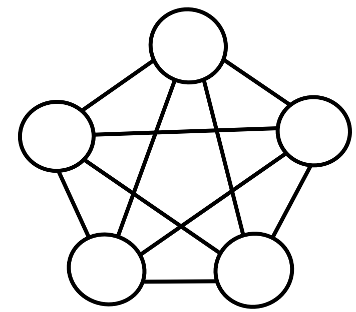
Coding with Side Information

Decentralized | Partial Information | Centralized

$$\text{Storage Cost} \geq \left(\frac{\boldsymbol{\nu}}{\boldsymbol{c}} - \frac{\nu(\nu-1)}{c^2} + o\left(\frac{1}{c^2}\right)\right)K$$



$$\text{Storage Cost} = \frac{\mathbf{1}}{\boldsymbol{c}}K$$

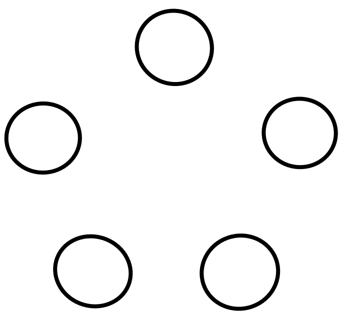


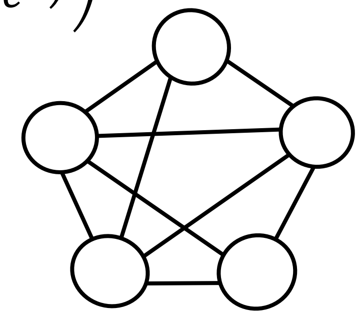
$$\text{Storage Cost} = \left(\frac{\mathbf{1}}{\boldsymbol{c}} + \frac{(\nu-1)\bar{m}(\mathcal{G})}{c^2} + o\left(\frac{\bar{m}(\mathcal{G})}{c^2}\right)\right)K$$

Coding with Side Information

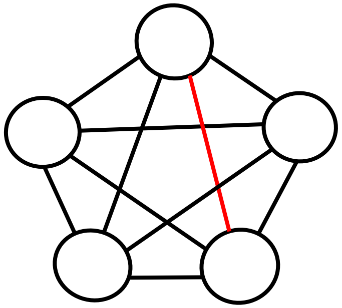
Decentralized | Partial Information | Centralized

$$\text{Storage Cost} \geq \left(\frac{\boldsymbol{\nu}}{\boldsymbol{c}} - \frac{\nu(\nu-1)}{c^2} + o\left(\frac{1}{c^2}\right)\right) K$$





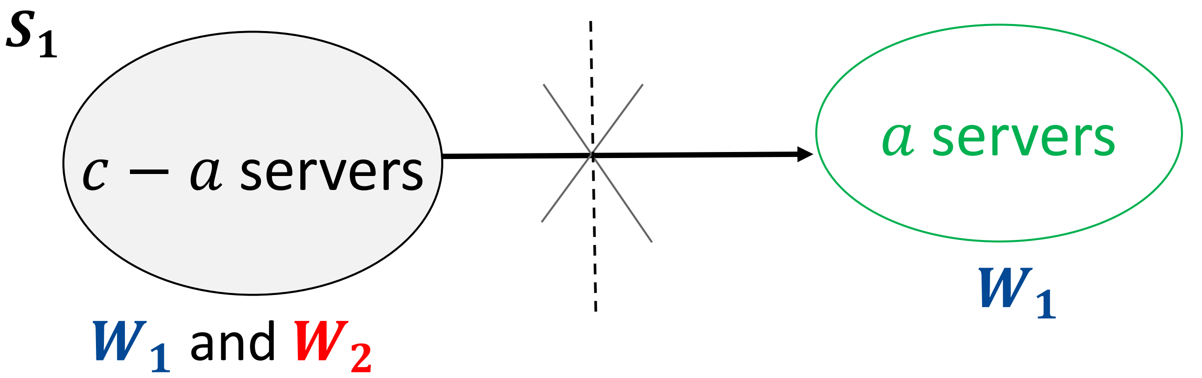
$$\text{Storage Cost} = \frac{\mathbf{1}}{\boldsymbol{c}} K$$



Storage Reduction $= 11\%$

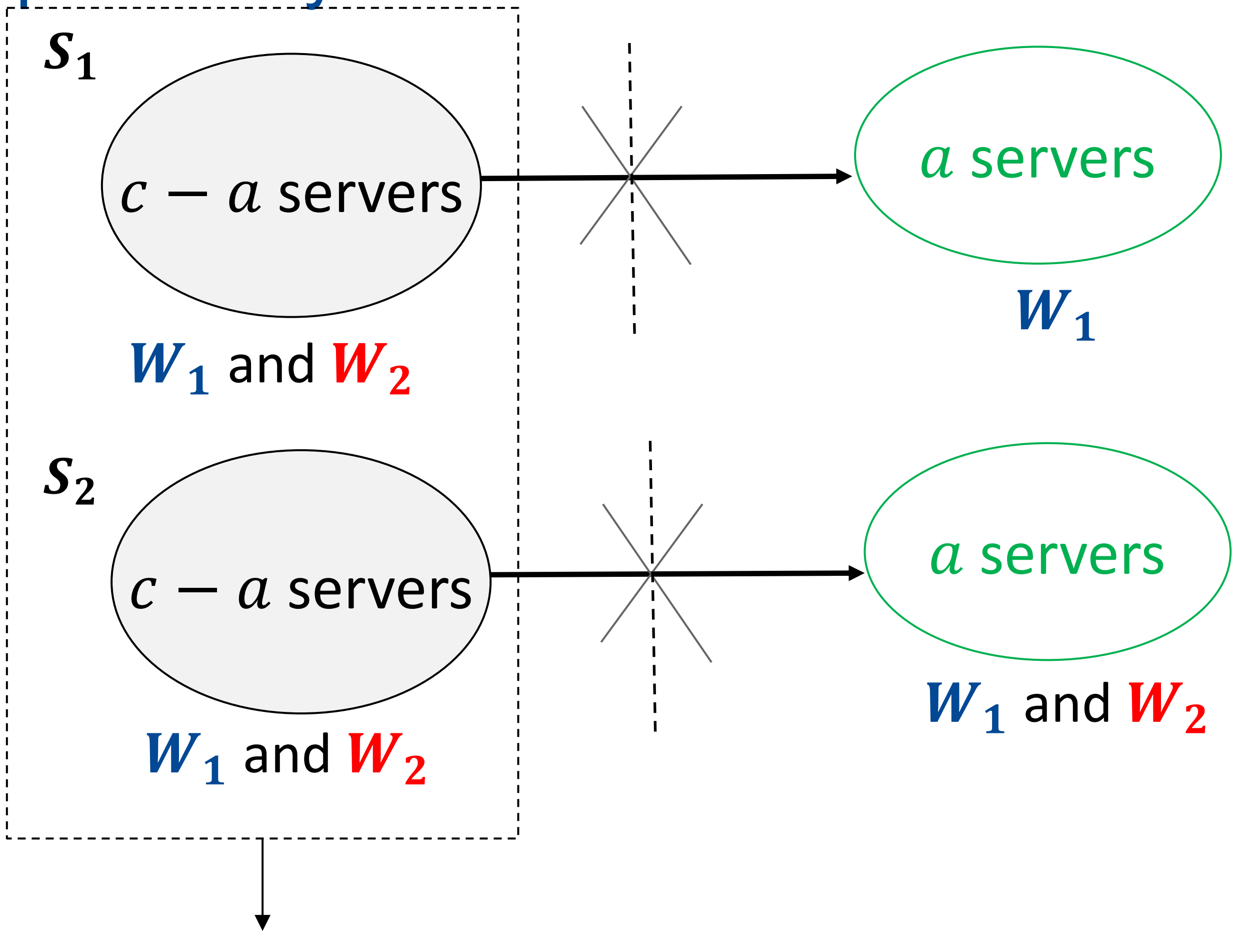
$$(n = 5, c_W = c_R = 4, \nu = 2)$$

Impossibility Results



W_1 is the latest complete version

Impossibility Results

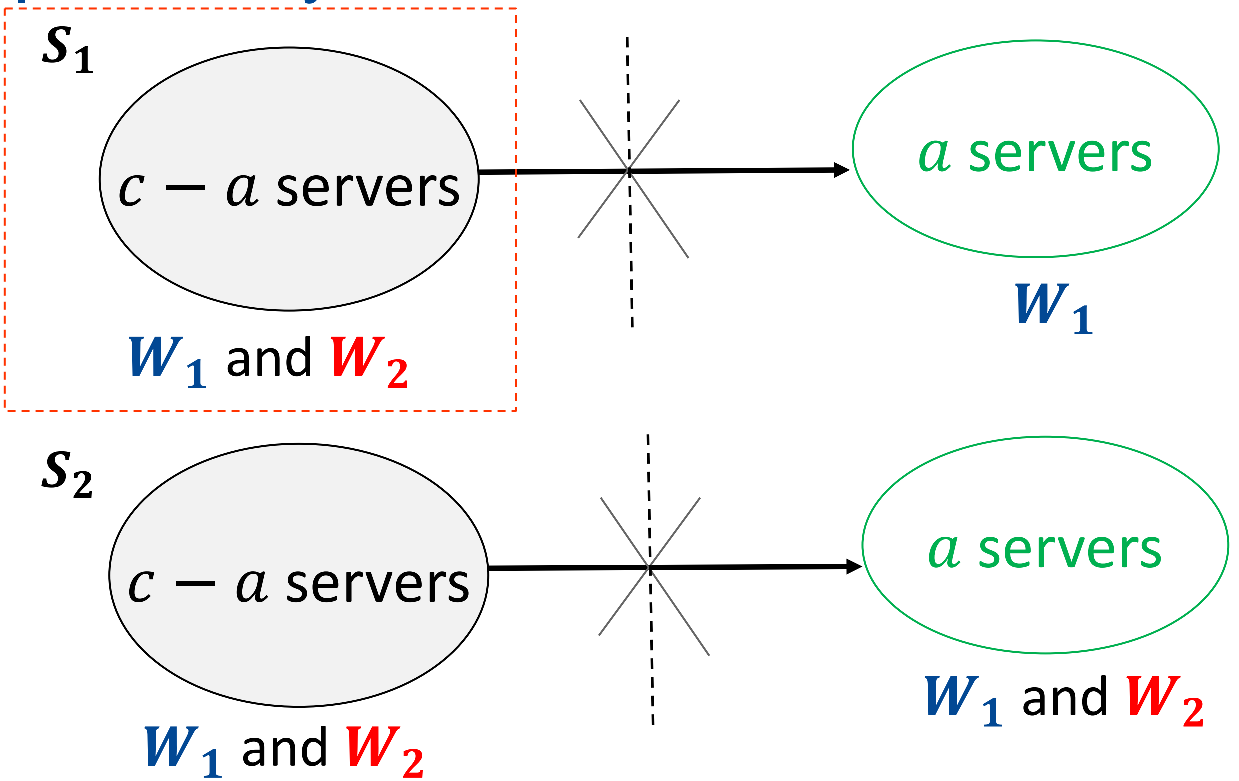


W_1 is the latest complete version

W_2 is the latest complete version

cannot differentiate between S_1 and S_2

Impossibility Results

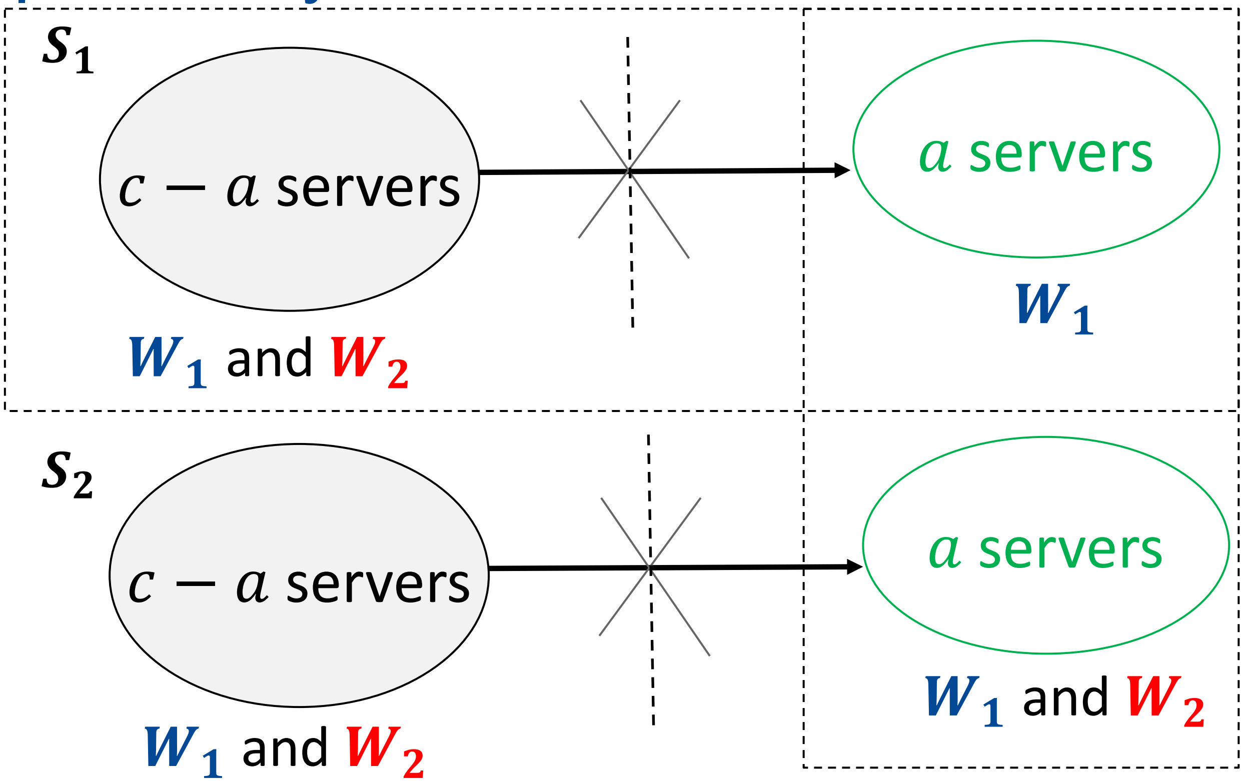


W_1 is the latest complete version

W_2 is the latest complete version

Decoding W_2 in S_1 $\Rightarrow$ $\text{Storage Cost} \geq \frac{1}{c-a} K$

Impossibility Results



W_1 is the latest complete version

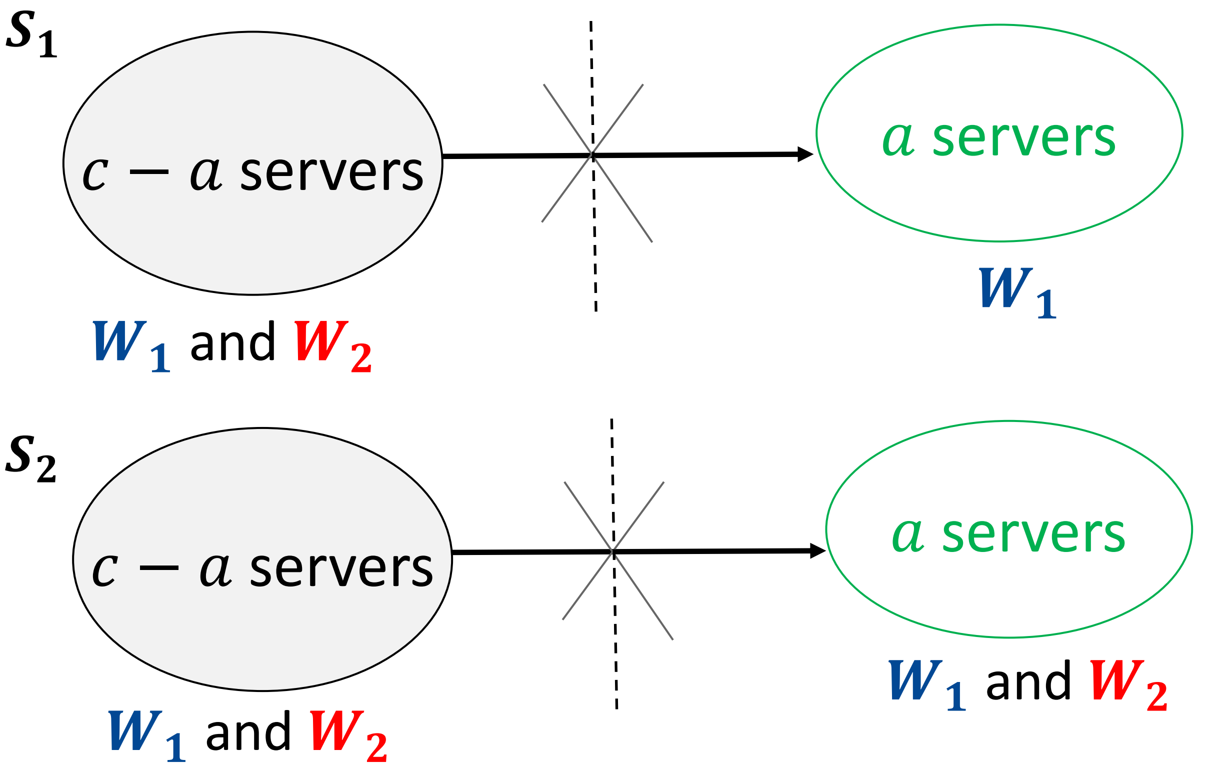
W_2 is the latest complete version

Decoding W_1 in S_1 ⇒ Storage Cost $\geq \frac{2}{c+a} K$

2 → W_1 in S_1, W_2 in S_2

$c+a$ → c servers of S_1 and the a servers of S_2

Impossibility Results



W_1 is the latest complete version

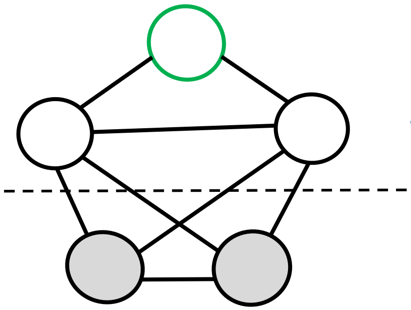
W_2 is the latest complete version

$$\text{Storage Cost} \geq \min\left\{\frac{1}{c-a}, \frac{2}{c+a}\right\} K$$

Impossibility Results

$$\text{Storage Cost} \geq \min\left\{\frac{1}{c-a}, \frac{2}{c+a}\right\} K$$

Implication:



Side Information is not useful

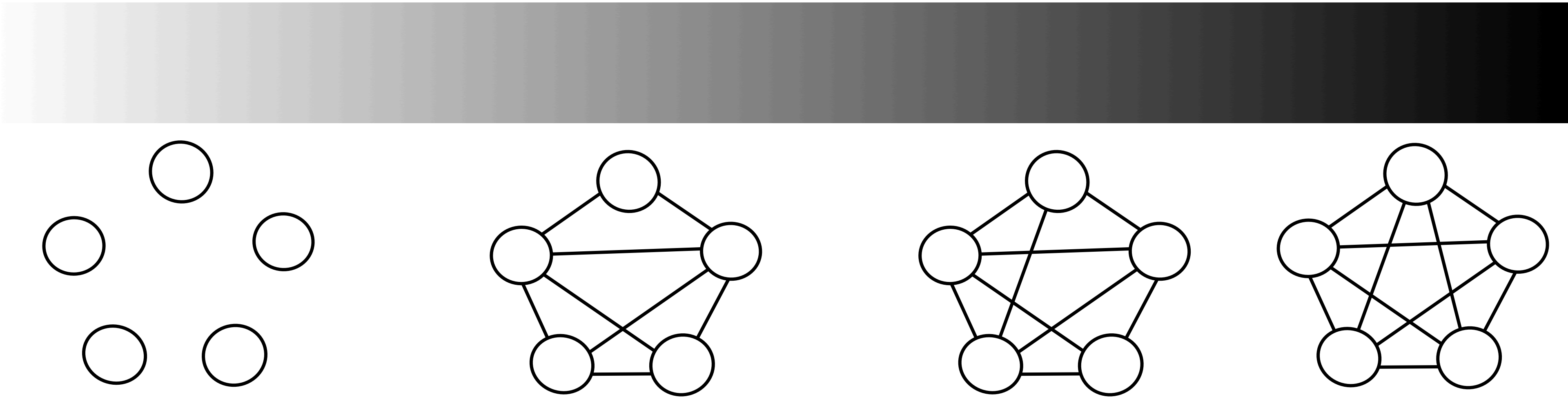
$(n = 5, c_W = c_R = 4, \nu = 2)$

$(c = 3, a = 1)$

Storage Cost $\geq K/2 \longrightarrow$ Can be achieved without side information [Wang et al. 2014]

Coding with Side Information

Decentralized Partial Information Centralized



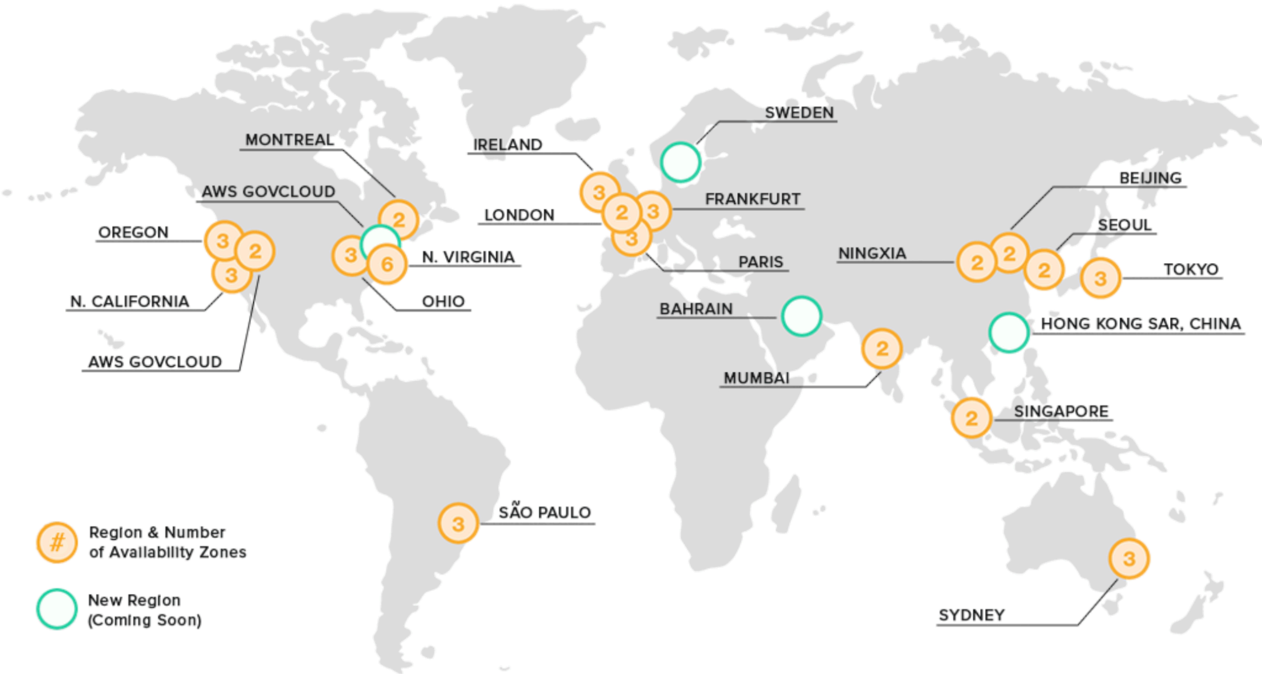
Side Information is not useful

Side Information is useful

$$(n = 5, c_W = c_R = 4, \nu = 2)$$

A careful study of the network topology is necessary

Case Study: Amazon Web Services (AWS)



Data center	Location	Data center	Location	Data center	Location
1	Tokyo	6	Frankfurt	11	Ohio
2	Seoul	7	Ireland	12	N. California
3	Mumbai	8	London	13	Oregon
4	Singapore	9	Paris		
5	Canada	10	N. Virginia		

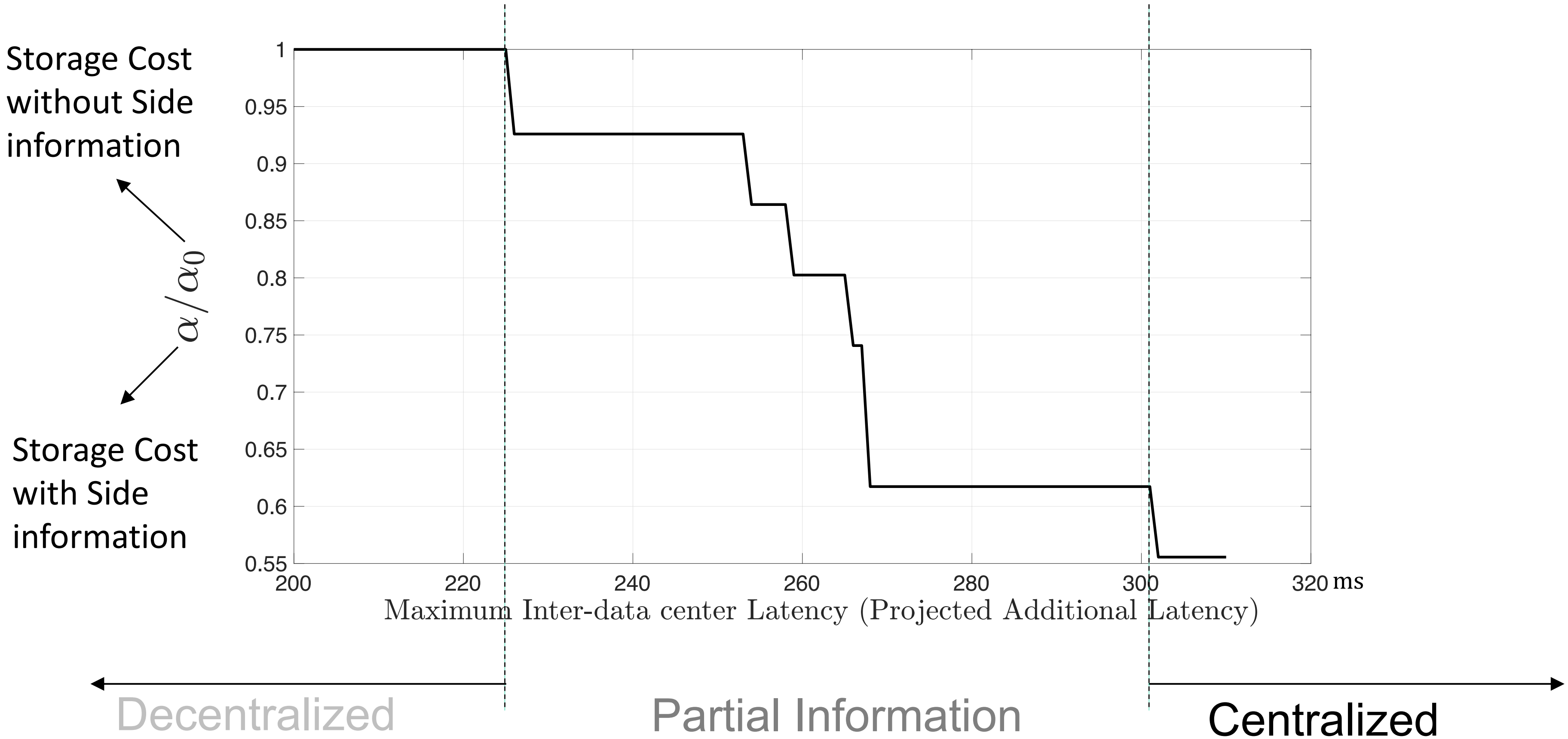
Case Study: AWS Inter-Region Latency

Data center	1	2	3	4	5	6	7	8	9	10	11	12	13
1	0	37.8	157.2	90.8	177.2	249.7	234.4	259.4	259.4	167.5	166.2	119.6	106.5
2	37.9	0	160.1	105.7	199.7	269.9	255.7	269.3	268.2	190.7	189.3	153	128.2
3	136.9	181.5	0	68.8	212.8	129.9	134.4	128	118.3	187.7	202.2	240.8	225
4	90	112.4	82.3	0	240.9	189.7	186.4	181.3	178.5	267.8	232.6	184.7	194.7
5	159.2	189.5	202	222.3	0	103.1	81.7	92	95.4	17.8	27.2	82	81.7
6	241.3	267.3	115.3	174.8	107	0	24.2	19.1	12.8	90.4	98.9	147.8	165.4
7	230	258.4	128.4	180	85.2	23.8	0	14.6	21.6	72.7	84.6	152.8	137.4
8	236.9	265.3	116.9	168	93.9	15.7	13.2	0	10.7	78	88.7	141.7	148.5
9	233.5	301.6	111.6	173	97.6	14.4	20.4	11	0	81.7	99.4	140.7	157.8
10	164.3	188.8	195.8	239.9	18.8	92	73.1	79.8	110.5	0	13.66	67.2	79.3
11	162.4	189.9	199.7	226	27.6	121.5	87.7	91.3	94.6	16.4	0	55.9	74.53
12	111.4	157.9	253.4	178.3	81.7	148.7	150.7	140	146.7	67.8	53.9	0	23.4
13	109.8	139.7	226	166.5	73.4	167.8	137.8	150.8	160.4	84	73	25.8	0

Source: https://www.cloudping.co/

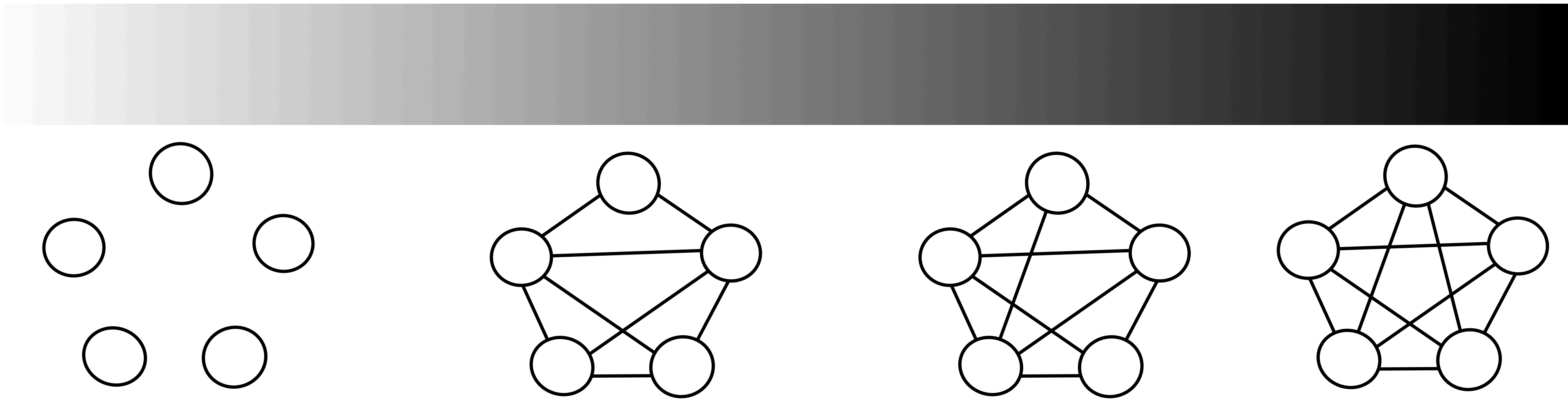
An edge exists between node i and node j if the latency between them $\leq$ maximum allowable latency

Case Study: Latency-Storage Trade-off in AWS



Discussion

Decentralized | Partial Information | Centralized



Side Information is not useful

Side Information is useful

$(n = 5, c_W = c_R = 4, \nu = 2)$

Questions?

Thank You