Distributed Function Computation over Networks

Derya Malak

Advanced topics in wireless communications
(AtWireless)

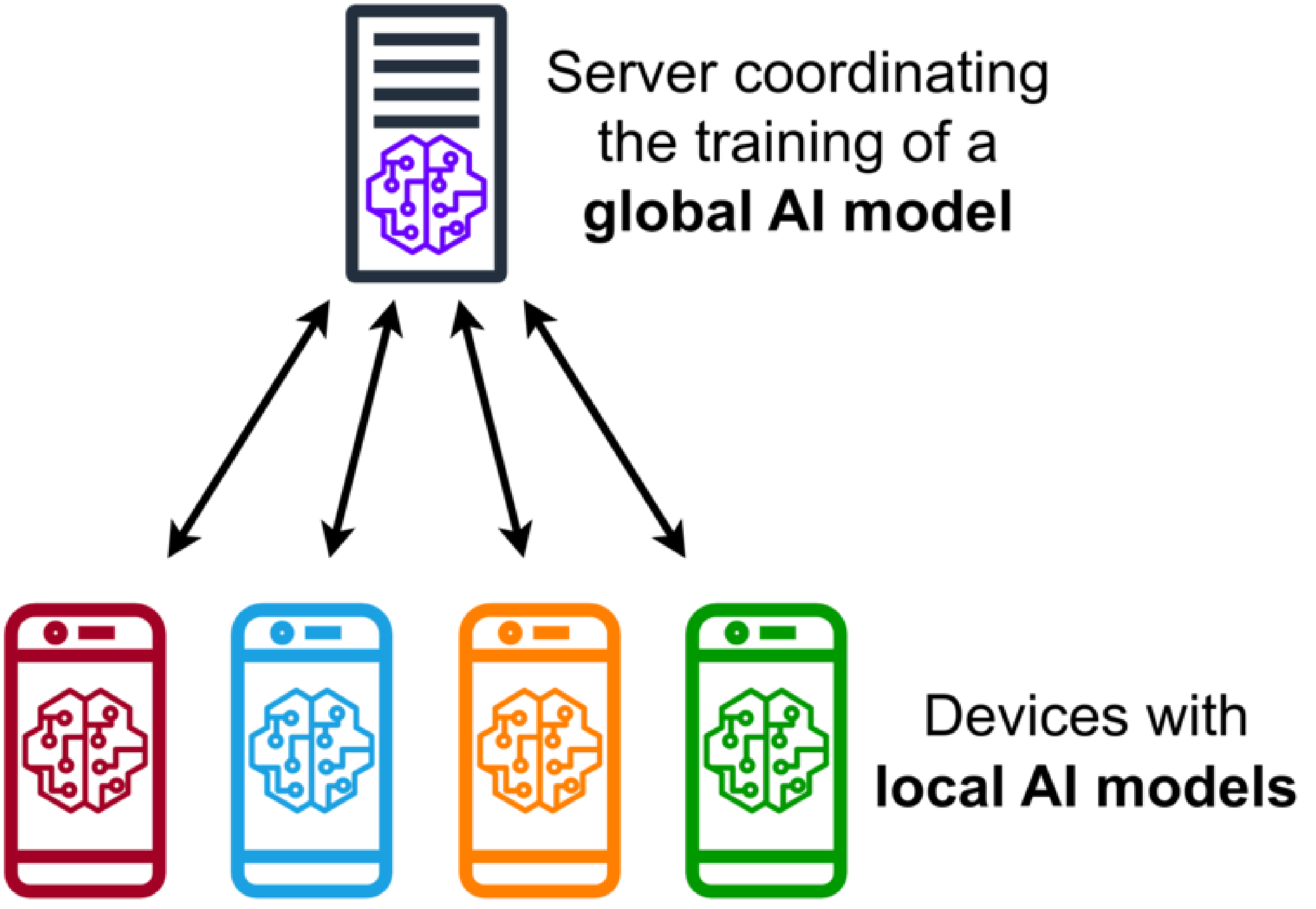
EURECOM
Sophia Antipolis

October 25, 2024

Lecture Outline

- **Part I. Distributed computation**
  - Motivation and challenges
  - Objectives
- **Part II. Techniques for distributed computing**
  - Information-theoretic (Distributed communication)
  - Algebraic (Körner-Marton, Krithivasan-Pradhan)
  - Graph-theoretic (Alon, Orlitsky, Roche, Médard)
  - Coding-theoretic
  - Communication-computation tradeoffs (Gradient coding, distributed matrix multiplication)
- **Part III. Exploiting structural properties**
  - Source (joint distribution, common information)
  - Function (separability, decomposability)
  - Network (topological structures)

Distributed Computing



(Wikipedia) a Federated Learning protocol with smartphones

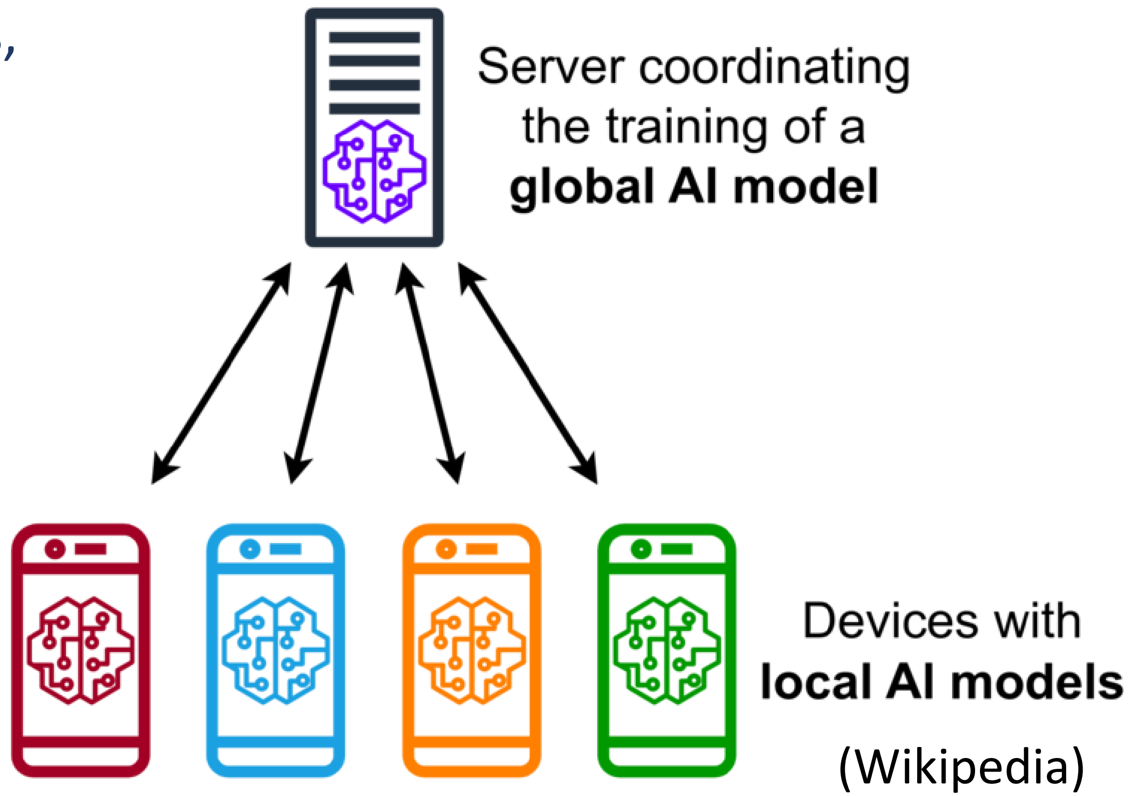
Distributed Computing

Distributed computing plays a key role in accelerating the execution of computationally challenging tasks via

- distributing workload across multiple servers, (reduces workload!)
- leveraging collective computational capabilities, (saves resources!)

&

- harnessing parallel processing to fulfill tasks (saves time!)



(Wikipedia)

Applications of Distributed Computing

Telecommunications (cellular networks, wireless sensor networks, routing algorithms)

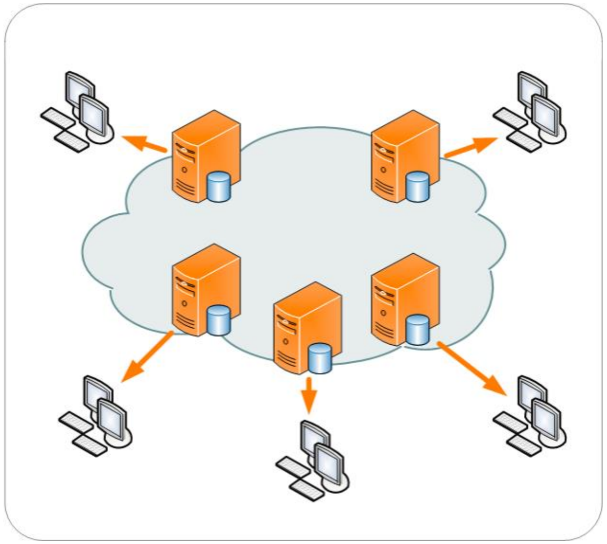
Networking (distributed databases, distributed caching systems, smart grid, Internet of Things)

Real-time process control (Industrial control systems, medical applications [Lushbough, Brendel, 2010], autonomous vehicles)

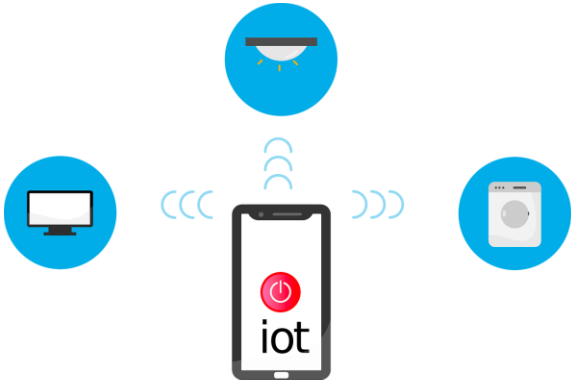
Parallel computation (scientific computing, cluster computing, cloud computing [Shamsi *et al.*, 2013], computer graphics [Gao, Wang, Zhou, 2019])

Peer-to-peer (blockchain [Bagaria *et al.*, 2019])

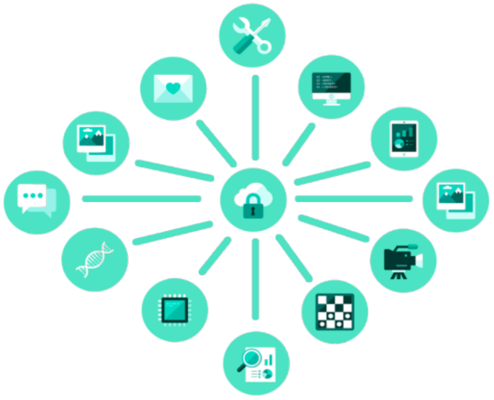
Applications of Distributed Computing



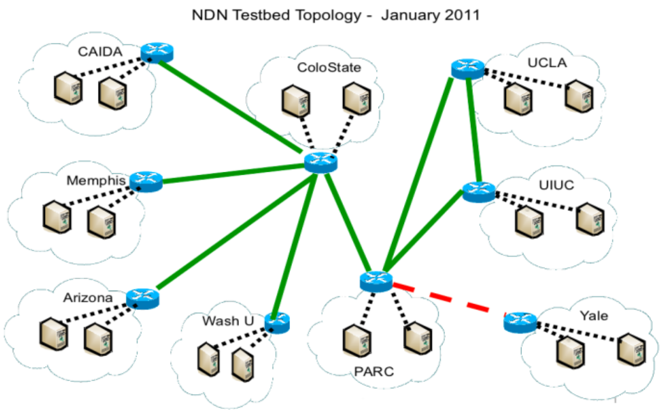
CDNs



Edge/Wireless
IoT

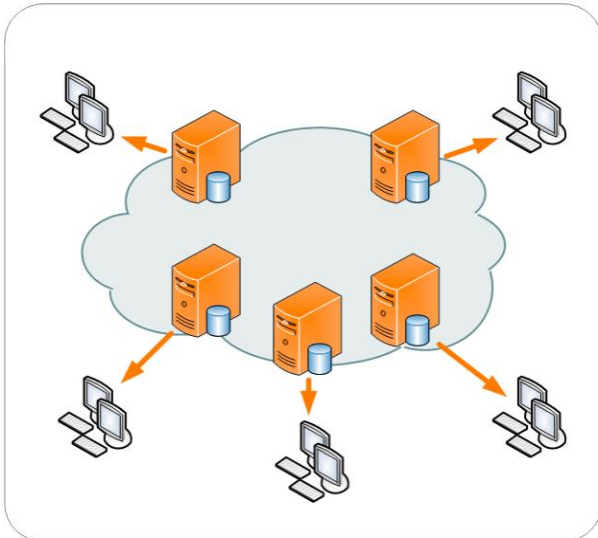


Cloud Computing

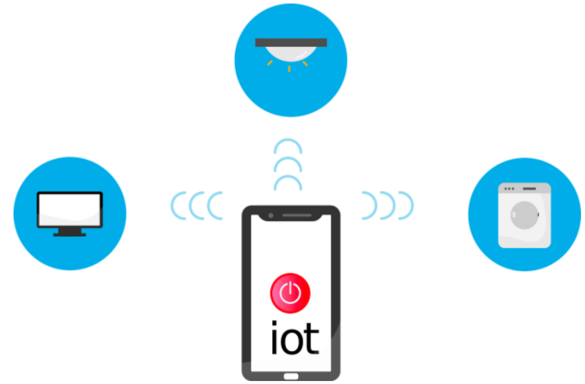


Content-Centric
Networking

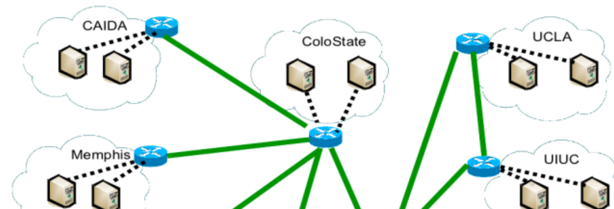
Applications of Distributed Computing



CDNs



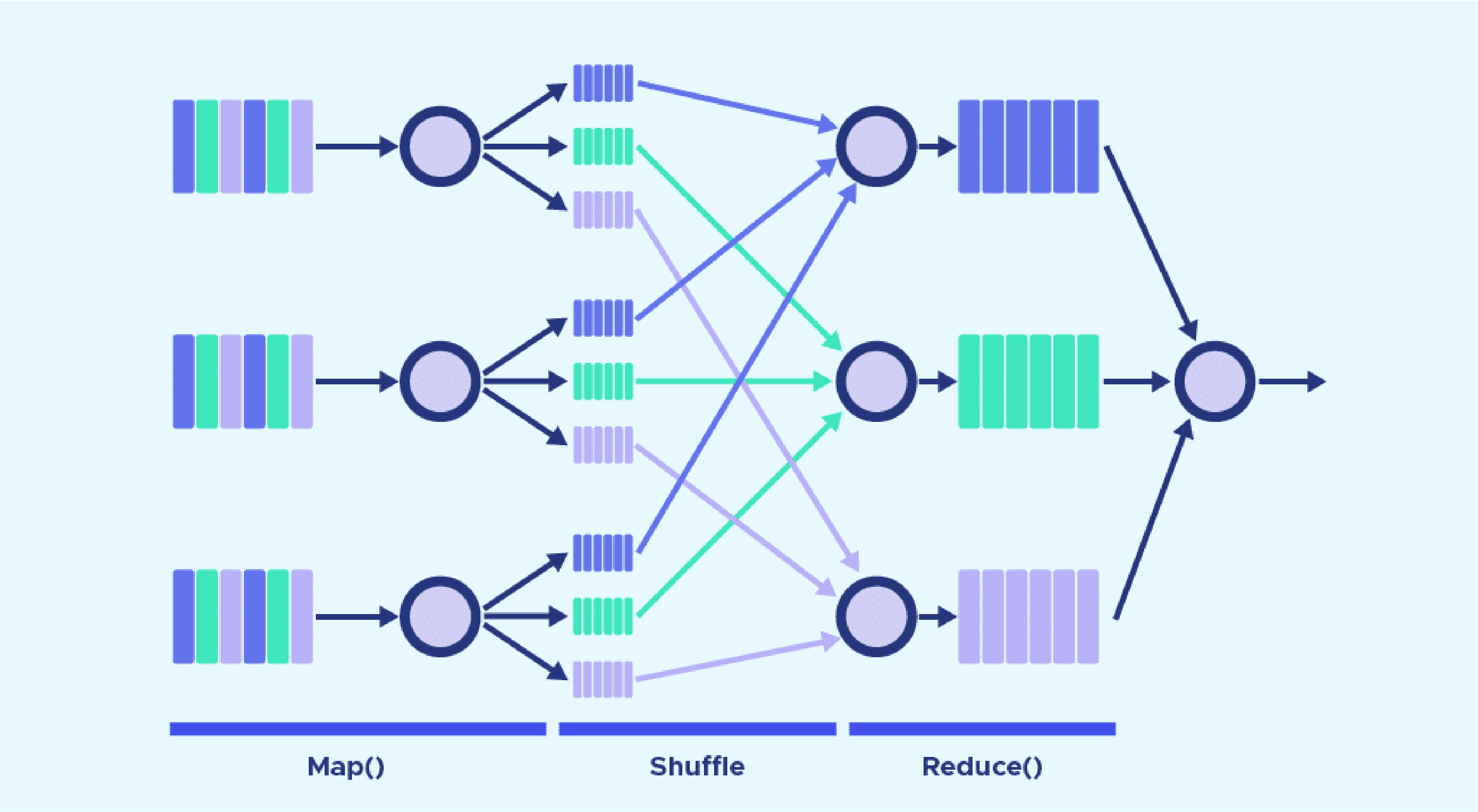
Edge/Wireless
IoT



Distributed computing can address a wide array of complex computational challenges and data-intensive analyses.

Large-Scale Distributed Computing

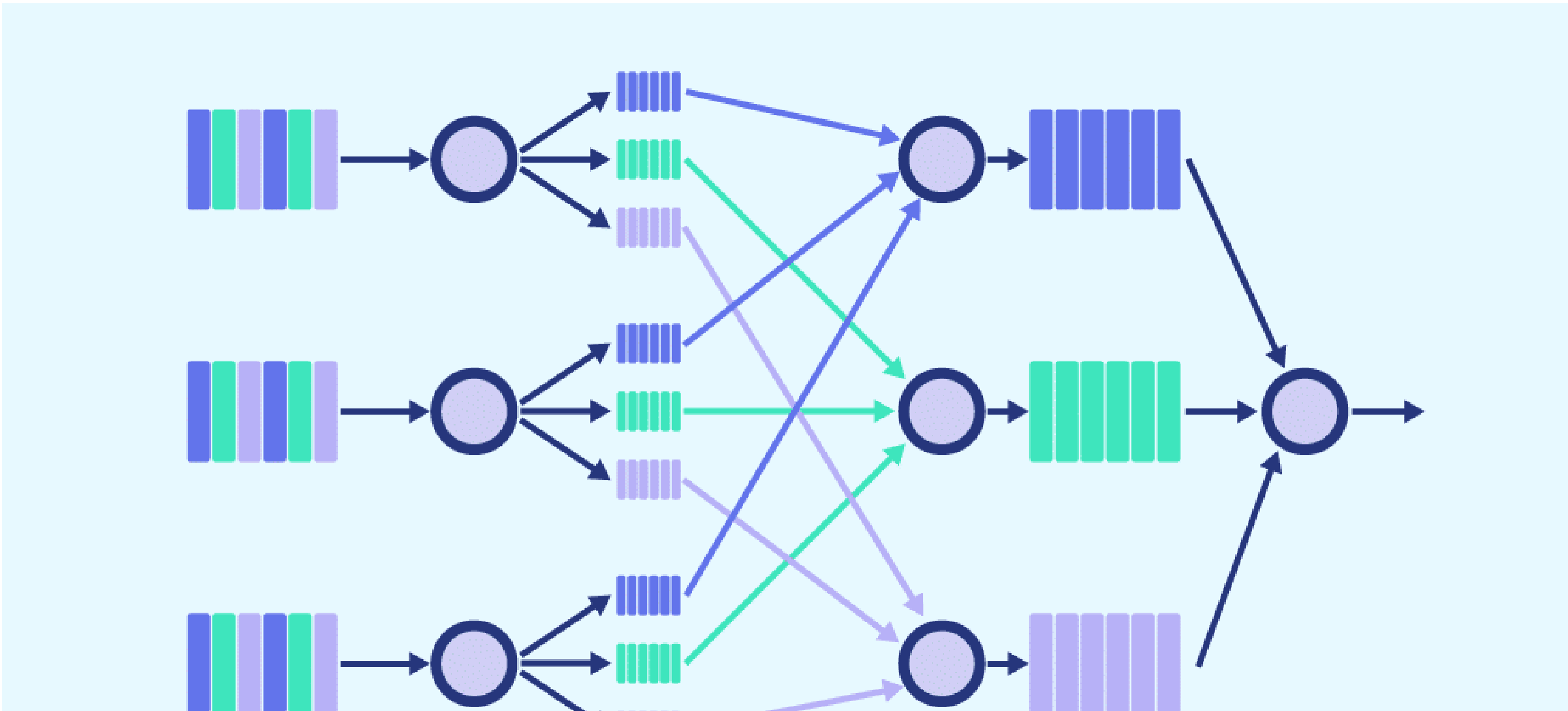
MapReduce, Spark, federated learning algorithms, or distributed deep networks, parallelize the execution of computations [Dikaiakos *et al.*, 2009]



https://datascientest.com/en/mapreduce-how-to-use-it-for-big-data

Large-Scale Distributed Computing

MapReduce, Spark, federated learning algorithms, or distributed deep networks, parallelize the execution of computations [Dikaiakos *et al.*, 2009]



The shift from centralized computation to distributed computing is inevitable because of the flourishing demands for scalability, efficiency, and performance.

Distributed computation models face severe challenges.

Accuracy [Jahani-Nezhad, Maddah-Ali, 2021], [Wang, Jia, Jafar, 2021],

Concurrency of components,

Privacy and **security** [Sun, Jafar, 2019] , [Soleymani, Mahdavifar, 2021],

Scalability of computing [Soleymani, Mahdavifar, Avestimehr, 2021] ,

Latency and **stragglers** [Li et al., 2021], **failure of components,**

Astronomical communication cost, often required by the distributed implementation of large-scale tasks,

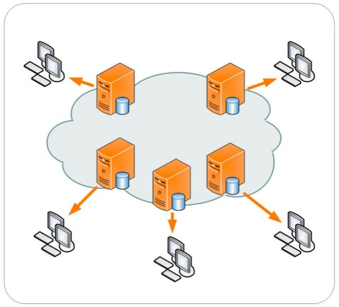
A struggle between **computation and communication complexity** lies at the heart of distributed computing.

Distributed computing: How to populate content in caches?

Any ideas?

How to allocate content?

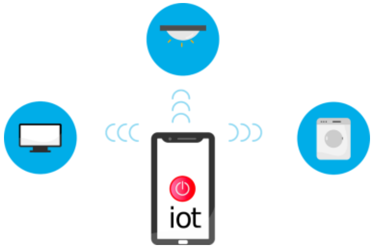
- Caching and content allocation problems are ubiquitous



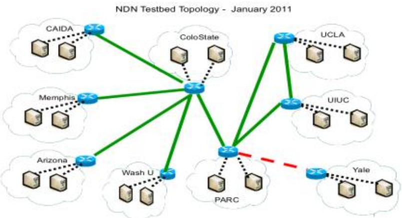
CDNs

Cloud Computing

- Placement problems are **combinatorial** [Shanmugam *et al.*, 2013]



Edge/Wireless IoT



Content-Centric Networking

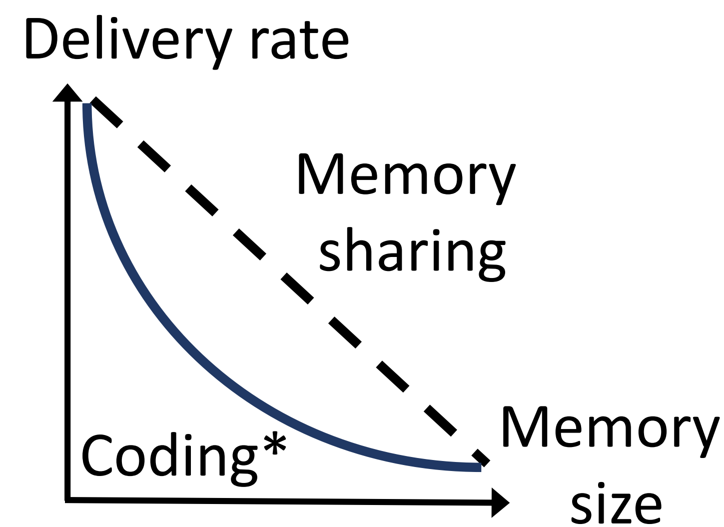
- **Coded caching** [Maddah-Ali-Niesen, 2014]
  - Relaxes combinatorial structure
  - Eases design/weakens constraints
  - Improves efficiency through cross-coding

Video is the Primary Bandwidth Hog in Wireless Systems

Video consumes **most of wireless bandwidth**

NETFLIX YouTube hulu f serve most of it!

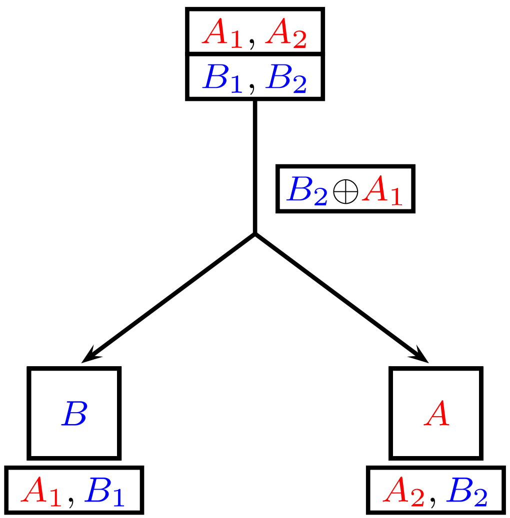
Caching helps **offload traffic** from congested networks



*Maddah-Ali, Niesen, "Fundamental limits of caching," *IEEE Trans. Inf. Theory*, May 2014.

Well-Known Caching Models

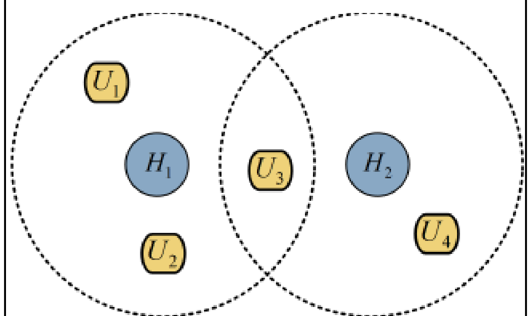
Fundamental limits [MAN14]



Caching strategy for M=2 files, K=2 users, cache size N = 1

Pro: multicasting
Con: strong assumptions

Femtocaching [SG13]

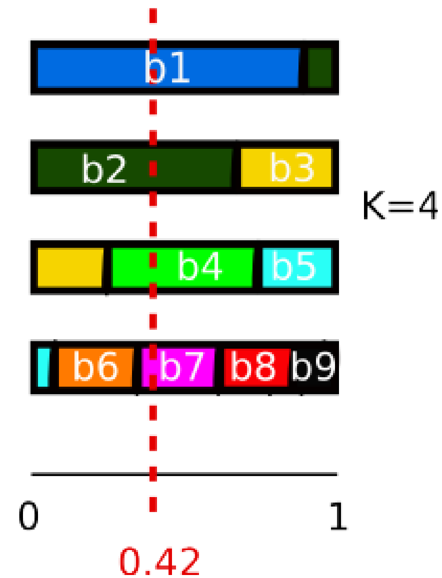


H_1, H_2: helpers

Distributed caching example: users with conflicting interests

Pro: 1-1/e factor
Con: computationally demanding

Geographic caching [BG15]

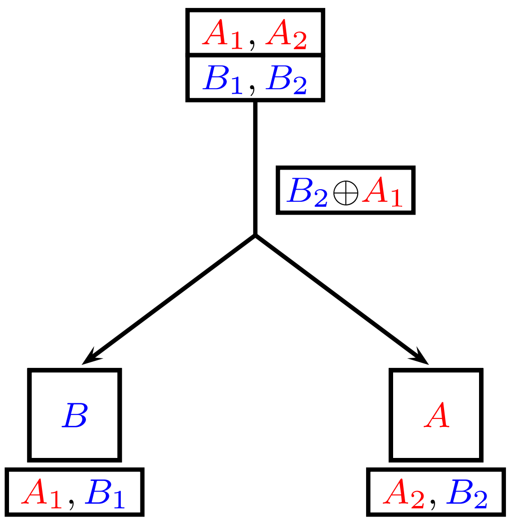


A realization of probabilistic placement

Pro: probabilistic
Con: does not exploit diversity

Well-Known Caching Models

Fundamental limits [MAN14]

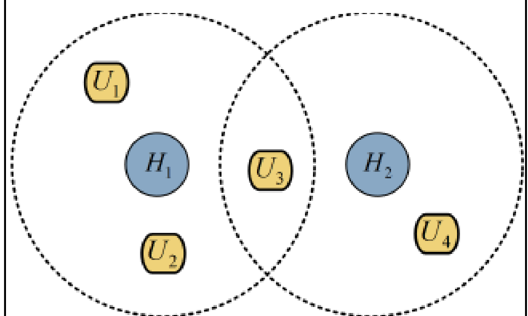


Caching strategy for M=2 files, K=2 users, cache si

Pro: mul

Con: str

Femtocaching [SG13]

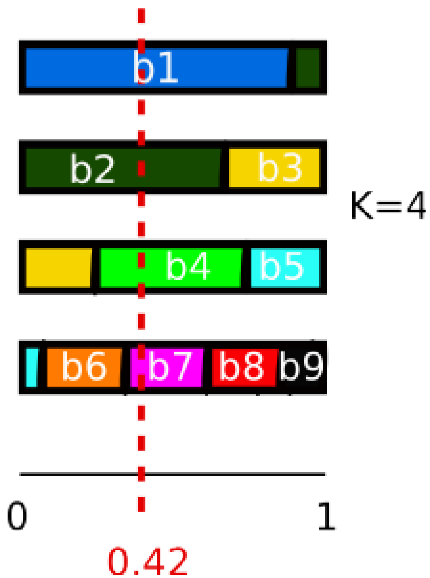


H_1, H_2: helpers

Distributed caching example: users with conflicting interests

Pro: 1-1/e factor

Geographic caching [BG15]



A realization of probabilistic placement

Pro: probabilistic

oit

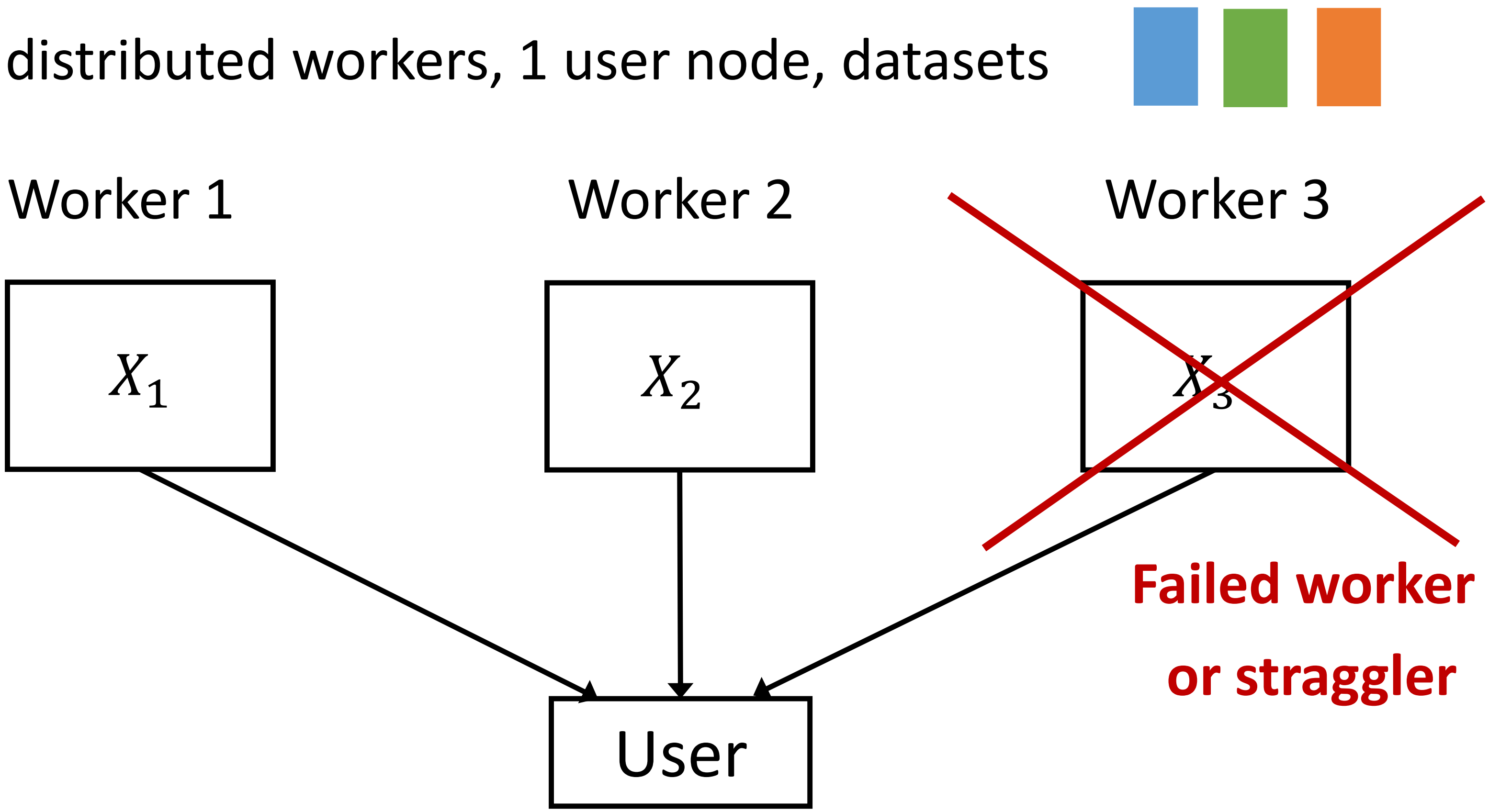
Goal: low complexity algorithms with performance guarantees

Distributed computing: How to mitigate stragglers?

Any ideas?

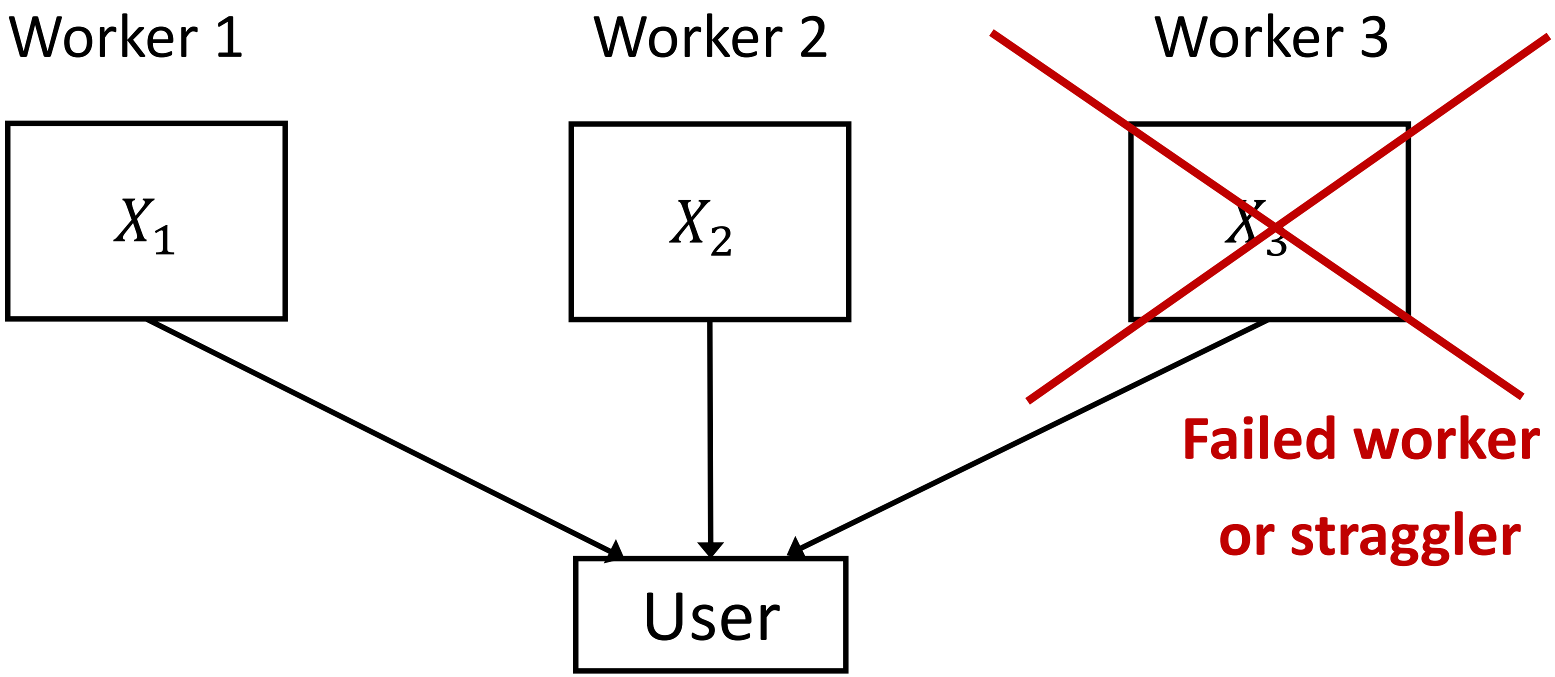
Distributed Function Computation

N=3 distributed workers, 1 user node, datasets



Distributed Function Computation

N=3 distributed workers, 1 user node, datasets



How to place the datasets so that the user can recover them from the remaining $N_r = 2$ workers?

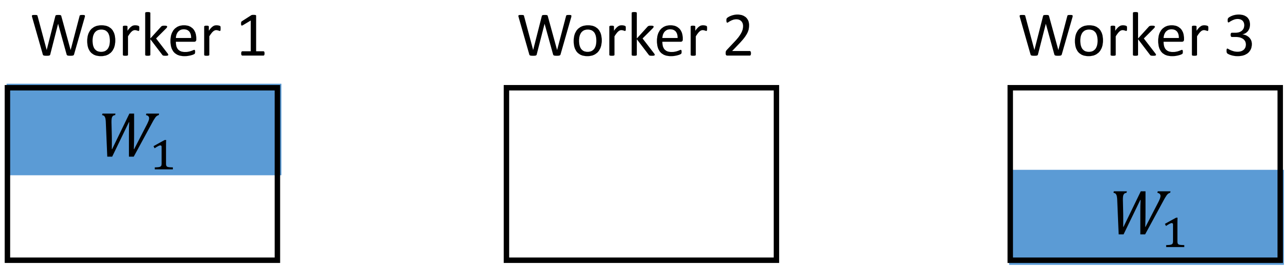
Dataset Placement Model

Example. $K = 3$ datasets $\{W_1, W_2, W_3\}$

$N = 3$ workers

$N_r = 2$ recovery threshold

$M = 2$ cache capacity



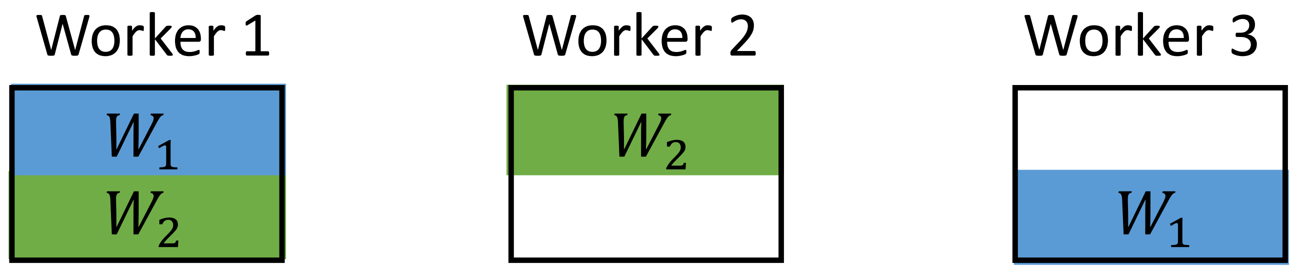
Dataset Placement Model

Example. $K = 3$ datasets $\{W_1, W_2, W_3\}$

$N = 3$ workers

$N_r = 2$ recovery threshold

$M = 2$ cache capacity



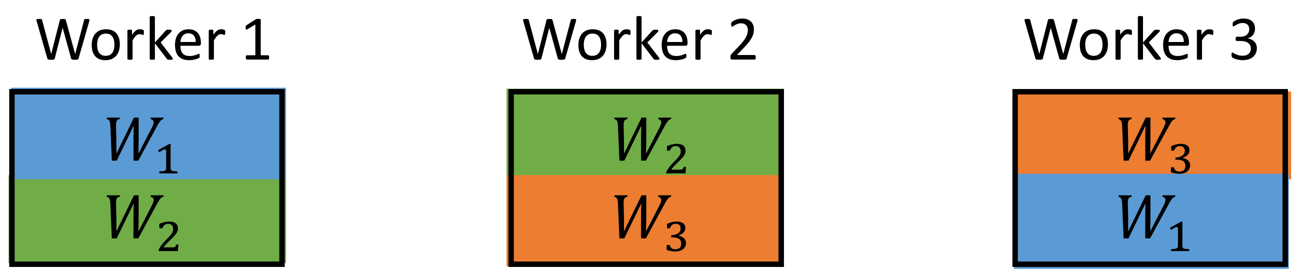
Dataset Placement Model

Example. $K = 3$ datasets $\{W_1, W_2, W_3\}$

$N = 3$ workers

$N_r = 2$ recovery threshold

$M = 2$ cache capacity



Dataset Placement Model

Example. $K = 3$ datasets $\{W_1, W_2, W_3\}$

$N = 3$ workers

$N_r = 2$ recovery threshold

$M = 2$ cache capacity

Worker 1	Worker 2	Worker 3
W_1	W_2	W_3
W_2	W_3	W_1
$X_1 = \{W_1, W_2\}$	$X_2 = \{W_2, W_3\}$	$X_3 = \{W_1, W_3\}$

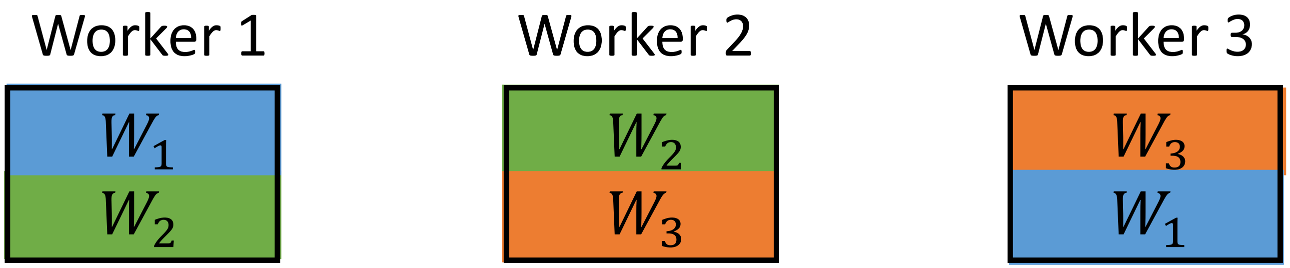
Dataset Placement Model

Example. $K = 3$ datasets $\{W_1, W_2, W_3\}$

$N = 3$ workers

$N_r = 2$ recovery threshold

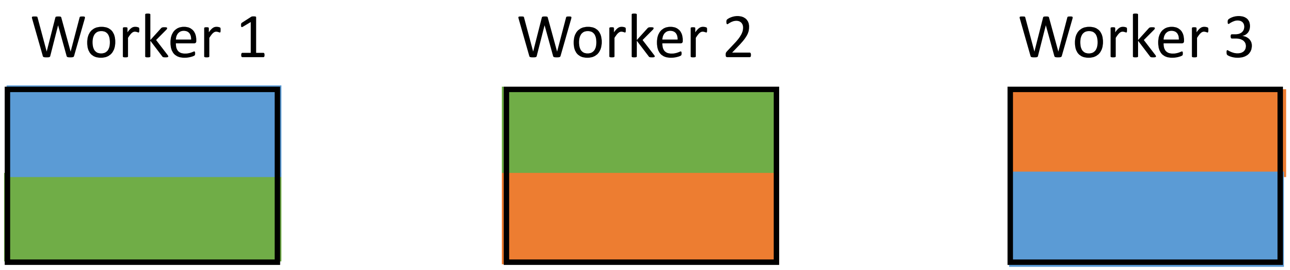
$M = 2$ cache capacity



$X_1 = \{W_1, W_2\}$ $X_2 = \{W_2, W_3\}$ $X_3 = \{W_1, W_3\}$

We can recover W_1, W_2, W_3 from any 2 workers.

Cyclic Dataset Placement Model



Cyclic symmetry: the set of datasets assigned to worker $i \in [N]$ is

$$\mathcal{Z}_i = \bigcup_{r \in \left[0:\frac{K}{N}-1\right]} \{ \mod\{i, N\} + rN, \quad \mod\{i+1, N\} + rN, \ldots, \mod\{i+N-N_r, N\} + rN \}$$

and $X_i = W_{\mathcal{Z}_i}$

Can we do better than this placement model?

Symmetry at a finer granularity

Example. $K = 3,\ N = 3,\ N_r = 2,\ M = 2$

Worker 1

$W_{1,\{1,2\}}$

$W_{1,\{1,3\}}$

$$X_1 = \{W_{k,\{1,2\}}, W_{k,\{1,3\}}\}_{k\in[3]}$$

Worker 2

$$X_2 = \{W_{k,\{1,2\}}, W_{k,\{2,3\}}\}_{k\in[3]}$$

Worker 3

$$X_3 = \{W_{k,\{1,3\}}, W_{k,\{2,3\}}\}_{k\in[3]}$$

[Maddah-Ali, Niesen, 2014] M. A. Maddah-Ali and U. Niesen, "Fundamental limits of caching," IEEE Trans. Inf. Theory, May 2014.

Symmetry at a finer granularity

Example. $K = 3,\ N = 3,\ N_r = 2,\ M = 2$

Worker 1

$W_{1,\{1,2\}}$	$W_{2,\{1,2\}}$
$W_{1,\{1,3\}}$	$W_{2,\{1,3\}}$

$$X_1 = \left\{W_{k,\{1,2\}}, W_{k,\{1,3\}}\right\}_{k\in[3]}$$

Worker 2

$$X_2 = \left\{W_{k,\{1,2\}}, W_{k,\{2,3\}}\right\}_{k\in[3]}$$

Worker 3

$$X_3 = \left\{W_{k,\{1,3\}}, W_{k,\{2,3\}}\right\}_{k\in[3]}$$

[Maddah-Ali, Niesen, 2014] M. A. Maddah-Ali and U. Niesen, “Fundamental limits of caching,” IEEE Trans. Inf. Theory, May 2014.

Symmetry at a finer granularity

Example. $K = 3,\ N = 3,\ N_r = 2,\ M = 2$

Worker 1

$W_{1,\{1,2\}}$	$W_{2,\{1,2\}}$	$W_{3,\{1,2\}}$
$W_{1,\{1,3\}}$	$W_{2,\{1,3\}}$	$W_{3,\{1,3\}}$

$$X_1 = \{W_{k,\{1,2\}}, W_{k,\{1,3\}}\}_{k\in[3]}$$

Worker 2

$$X_2 = \{W_{k,\{1,2\}}, W_{k,\{2,3\}}\}_{k\in[3]}$$

Worker 3

$$X_3 = \{W_{k,\{1,3\}}, W_{k,\{2,3\}}\}_{k\in[3]}$$

[Maddah-Ali, Niesen, 2014] M. A. Maddah-Ali and U. Niesen, “Fundamental limits of caching,” IEEE Trans. Inf. Theory, May 2014.

Symmetry at a finer granularity

Example. $K = 3,\ N = 3,\ N_r = 2,\ M = 2$

Worker 1

$W_{1,\{1,2\}}$	$W_{2,\{1,2\}}$	$W_{3,\{1,2\}}$
$W_{1,\{1,3\}}$	$W_{2,\{1,3\}}$	$W_{3,\{1,3\}}$

$$X_1 = \{W_{k,\{1,2\}}, W_{k,\{1,3\}}\}_{k\in[3]}$$

Worker 2

$W_{1,\{1,2\}}$	$W_{2,\{1,2\}}$	$W_{3,\{1,2\}}$

$$X_2 = \{W_{k,\{1,2\}}, W_{k,\{2,3\}}\}_{k\in[3]}$$

Subitem $W_{1,\{1,2\}}$ is stored in 2 workers.

Worker 3

$W_{1,\{1,3\}}$	$W_{2,\{1,3\}}$	$W_{3,\{1,3\}}$

$$X_3 = \{W_{k,\{1,3\}}, W_{k,\{2,3\}}\}_{k\in[3]}$$

Placement in a cyclic manner

[Maddah-Ali, Niesen, 2014] M. A. Maddah-Ali and U. Niesen, “Fundamental limits of caching,” IEEE Trans. Inf. Theory, May 2014.

Symmetry at a finer granularity

Example. $K = 3,\ N = 3,\ N_r = 2,\ M = 2$

Worker 1

$W_{1,\{1,2\}}$	$W_{2,\{1,2\}}$	$W_{3,\{1,2\}}$
$W_{1,\{1,3\}}$	$W_{2,\{1,3\}}$	$W_{3,\{1,3\}}$

$$X_1 = \{W_{k,\{1,2\}}, W_{k,\{1,3\}}\}_{k\in[3]}$$

Worker 2

$W_{1,\{1,2\}}$	$W_{2,\{1,2\}}$	$W_{3,\{1,2\}}$
$W_{1,\{2,3\}}$	$W_{2,\{2,3\}}$	$W_{3,\{2,3\}}$

$$X_2 = \{W_{k,\{1,2\}}, W_{k,\{2,3\}}\}_{k\in[3]}$$

Worker 3

$W_{1,\{1,3\}}$	$W_{2,\{1,3\}}$	$W_{3,\{1,3\}}$
$W_{1,\{2,3\}}$	$W_{2,\{2,3\}}$	$W_{3,\{2,3\}}$

$$X_3 = \{W_{k,\{1,3\}}, W_{k,\{2,3\}}\}_{k\in[3]}$$

Placement in a cyclic manner

[Maddah-Ali, Niesen, 2014] M. A. Maddah-Ali and U. Niesen, "Fundamental limits of caching," IEEE Trans. Inf. Theory, May 2014.

Symmetry at a finer granularity

Example. $K = 3,\ N = 3,\ N_r = 2,\ M = 2$

Worker 1

$W_{1,\{1,2\}}$	$W_{2,\{1,2\}}$	$W_{3,\{1,2\}}$
$W_{1,\{1,3\}}$	$W_{2,\{1,3\}}$	$W_{3,\{1,3\}}$

$X_1 = \{W_{k,\{1,2\}}, W_{k,\{1,3\}}\}_{k\in[3]}$

Worker 2

$W_{1,\{1,2\}}$	$W_{2,\{1,2\}}$	$W_{3,\{1,2\}}$
$W_{1,\{2,3\}}$	$W_{2,\{2,3\}}$	$W_{3,\{2,3\}}$

$X_2 = \{W_{k,\{1,2\}}, W_{k,\{2,3\}}\}_{k\in[3]}$

Placement in a cyclic manner

Worker 3

$W_{1,\{1,3\}}$	$W_{2,\{1,3\}}$	$W_{3,\{1,3\}}$
$W_{1,\{2,3\}}$	$W_{2,\{2,3\}}$	$W_{3,\{2,3\}}$

$X_3 = \{W_{k,\{1,3\}}, W_{k,\{2,3\}}\}_{k\in[3]}$

We recover $W_1 = (W_{1,\{1,2\}}, W_{1,\{1,3\}}, W_{1,\{2,3\}})$ from any 2 workers.

Trans. Inf. Theory, May 2014.

Dataset Placement Model

Symmetry at finer granularity [Maddah-Ali-Niesen, 2014]**:** each $W_k, k \in [K]$ is split into $\binom{N}{|\tau|}$ disjoint subitems of equal size:

$$W_k = (W_{k,\tau} : \tau \subset [N], |\tau| = N\gamma)$$

(e.g., $W_1 = (W_{1,\{1,2\}}, W_{1,\{1,3\}}, W_{1,\{2,3\}})$

The worker assignments $\mathrm{X}_i,\ i \in [N]$ are as follows

$$X_i = \{W_{k,\tau} : \tau \ni i, \quad \tau \subset [N], \quad |\tau| = \gamma N, \quad k \in [K]\}$$

Each cache memory unit is split into smaller chunks to ensure symmetry at a finer granularity.

[Maddah-Ali, Niesen, 2014] M. A. Maddah-Ali and U. Niesen, “Fundamental limits of caching,” IEEE Trans. Inf. Theory, May 2014.

Distributed computing: How to compute $f(X_1, X_2) = (X_1, X_2)$?

Any ideas?

Distributed source coding (distributed communication)

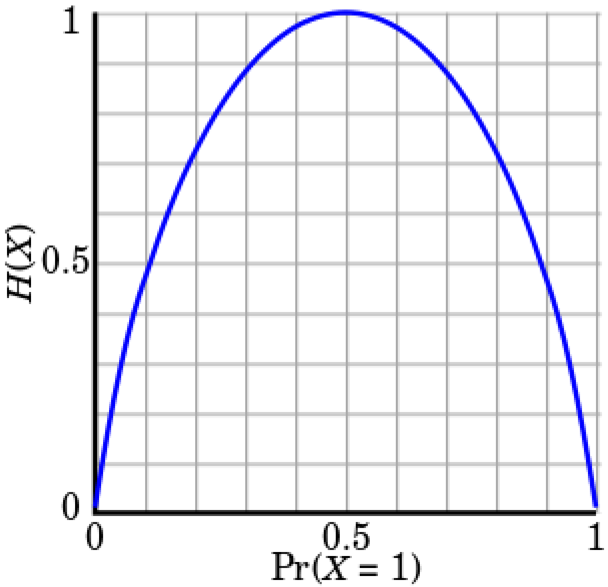
Some Definitions

Entropy (in bits)

$$H(X) = E[-log_2(X)] = -\sum_i p_i \, log_2 \, p_i$$

Binary entropy (in bits)

$$h(p) = -p log_2(p) - (1-p) \, log_2(1-p)$$



Some Definitions

Entropy (in bits)

$$H(X) = E[-\,log_2(X)] = -\sum_i p_i\, log_2\, p_i$$

Binary entropy (in bits)

$$h(p) = -plog_2(p) - (1-p)\, log_2(1-p)$$

Joint entropy

$$H(X_1\,, X_2) = H(X_1) + H(X_2 \mid X_1)$$

Some Definitions

Entropy (in bits)

$$H(X) = E[-\,log_2(X)] = -\sum_i p_i\, log_2\, p_i$$

Binary entropy (in bits)

$$h(p) = -p log_2(p) - (1-p)\, log_2(1-p)$$

Joint entropy

$$H(X_1\,, X_2) = H(X_1) + H(X_2 \mid X_1)$$

Mutual information

$$I(X_1\,; X_2) = H(X_1) - H(X_1 \mid X_2)$$

Doubly Symmetric Binary Source (DSBS)

A relationship **between jointly distributed binary sources**

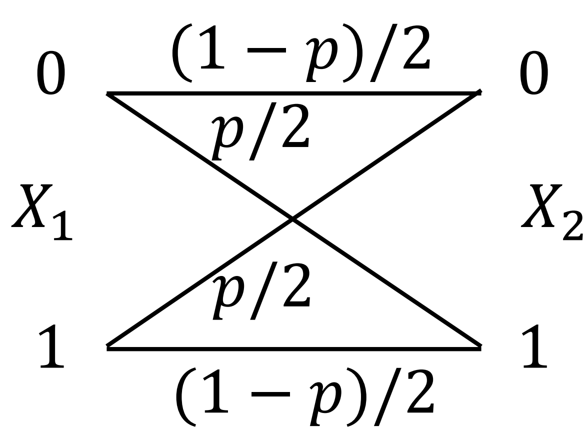
P_{X_1,X_2}		
	$P_{X_1,X_2}(0,0) = \frac{1-p}{2}$	$P_{X_1,X_2}(0,1) = \frac{p}{2}$
	$P_{X_1,X_2}(1,0) = \frac{p}{2}$	$P_{X_1,X_2}(1,1) = \frac{1-p}{2}$

P_{X_1,X_2} : joint distribution

DSBS

A relationship **between jointly distributed binary sources**

P_{X_1,X_2}	$P_{X_2}(0) = \frac{1}{2}$	$P_{X_2}(1) = \frac{1}{2}$
$P_{X_1}(0) = \frac{1}{2}$	$P_{X_1,X_2}(0,0) = \frac{1-p}{2}$	$P_{X_1,X_2}(0,1) = \frac{p}{2}$
$P_{X_1}(1) = \frac{1}{2}$	$P_{X_1,X_2}(1,0) = \frac{p}{2}$	$P_{X_1,X_2}(1,1) = \frac{1-p}{2}$



DSBS

A relationship **between jointly distributed binary sources**

P_{X_1,X_2}	$P_{X_2}(0) = \frac{1}{2}$	$P_{X_2}(1) = \frac{1}{2}$
$P_{X_1}(0) = \frac{1}{2}$	$P_{X_1,X_2}(0,0) = \frac{1-p}{2}$	$P_{X_1,X_2}(0,1) = \frac{p}{2}$
$P_{X_1}(1) = \frac{1}{2}$	$P_{X_1,X_2}(1,0) = \frac{p}{2}$	$P_{X_1,X_2}(1,1) = \frac{1-p}{2}$

$$H(X_1) = 1$$
$$H(X_2 \mid X_1) = H(X_1 \mid X_2) = h(p)$$
$$H(X_1, X_2) = H(X_1) + H(X_2 \mid X_1) = 1 + h(p)$$
$$I(X_1; X_2) = H(X_1) - H(X_1 \mid X_2) = 1 - h(p)$$

DSBS

A relationship **between jointly distributed binary sources**

P_{X_1,X_2}	$P_{X_2}(0) = \frac{1}{2}$	$P_{X_2}(1) = \frac{1}{2}$
$P_{X_1}(0) = \frac{1}{2}$	$P_{X_1,X_2}(0,0) = \frac{1-p}{2}$	$P_{X_1,X_2}(0,1) = \frac{p}{2}$
$P_{X_1}(1) = \frac{1}{2}$	$P_{X_1,X_2}(1,0) = \frac{p}{2}$	$P_{X_1,X_2}(1,1) = \frac{1-p}{2}$

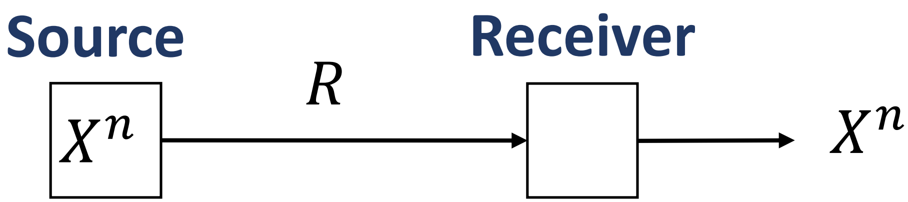
$$\begin{aligned} H(X_1, X_2) &= H(X_1) + H(X_2 \mid X_1) = 1 + h(p) \\ &\le H(X_1) + H(X_2) = 1 + 1 = 2 \end{aligned}$$

$H(X_1, X_2)$ is the minimum rate at which the sources can be jointly compressed.

Achievable Rate in Point to Point Communication

Key idea: consider long sequences

$$X^n = X_1, X_2, \dots . X_n \text{ (as } n \text{ goes to infinity)}$$



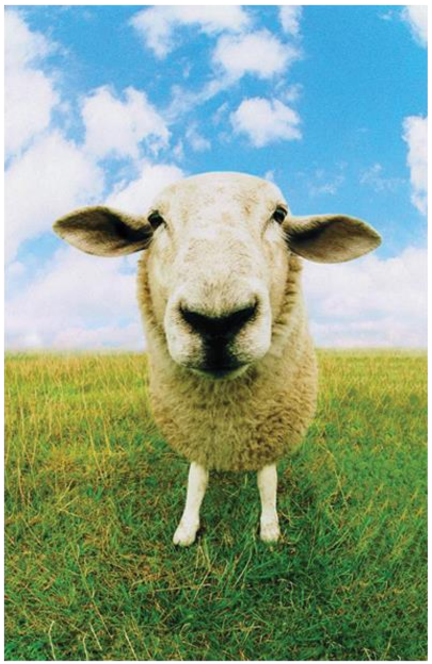
$$R \geq H(X)$$

Practical techniques: Huffman coding, Lempel-Ziv coding.

[Shannon, 1948] C. E. Shannon, "A mathematical theory of communication", Bell System Technical Journal, 1948.

Rate-Distortion Function

Communications with a fidelity constraint [Shannon, 1948]



High Resolution

Source

Receiver

$X^n \xrightarrow{R(D^*)} \square \longrightarrow \hat{X}^n$

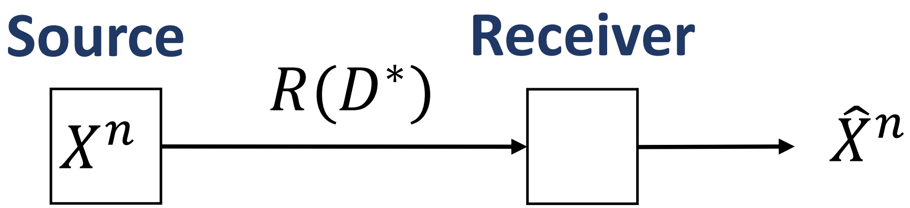
D^* is the maximum average distortion

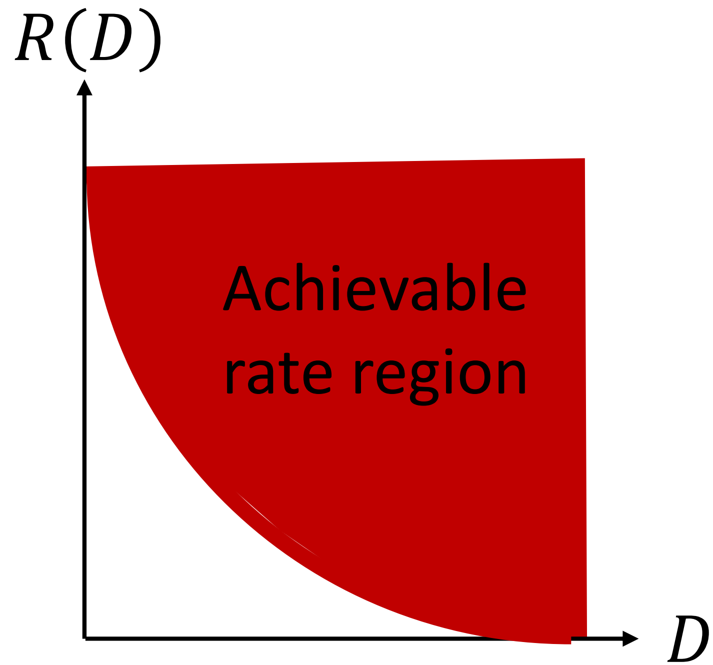
Low Resolution

[Shannon, 1948] C. E. Shannon, "A mathematical theory of communication", Bell System Technical Journal, 1948.

Rate-Distortion Function

Communications with a fidelity constraint [Shannon, 1948]



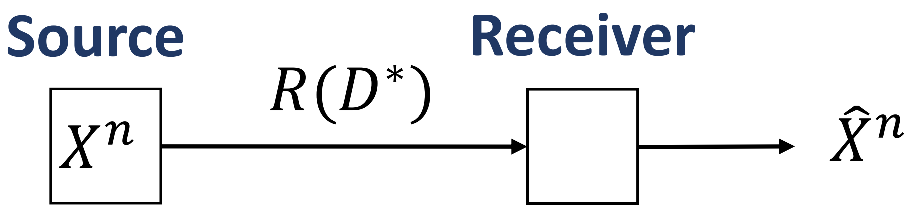


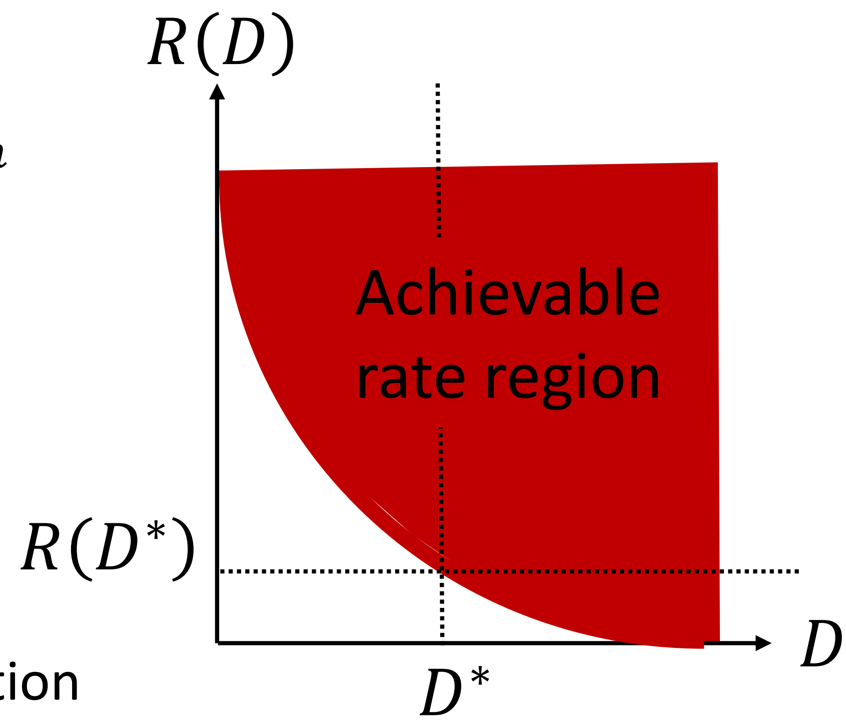
$$R(D^*) \geq \inf_{D \leq D^*} I(X; \hat{X})$$

[Shannon, 1948] C. E. Shannon, "A mathematical theory of communication", Bell System Technical Journal, 1948.

Rate-Distortion Function

Communications with a fidelity constraint [Shannon, 1948]





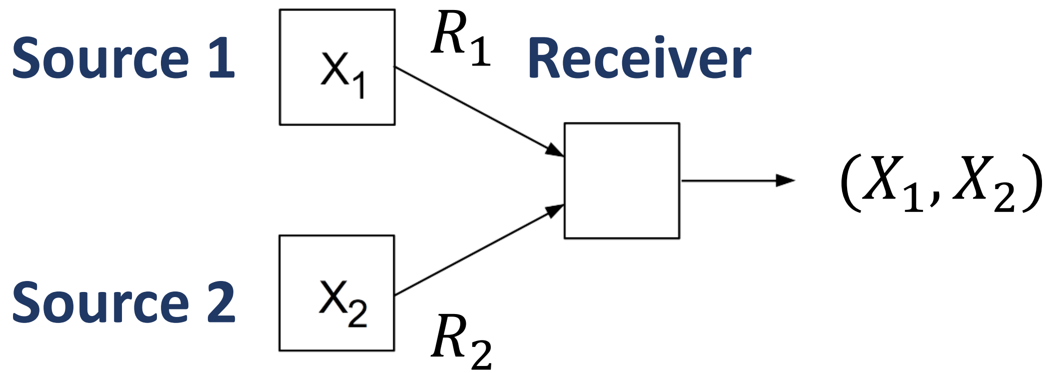
$$R(D^*) \geq \inf_{D \leq D^*} I(X;\hat{X})$$

maximum
average distortion

[Shannon, 1948] C. E. Shannon, “A mathematical theory of communication”, Bell System Technical Journal, 1948.

Computing $f(X_1, X_2) = (X_1, X_2)$

Distributed source compression [Slepian-Wolf, 1973]



Consider source sequences

$$X_j^n = X_{j1}, X_{j2}, \dots . X_{jn} \quad \text{(as } n \to \text{infinity)}$$

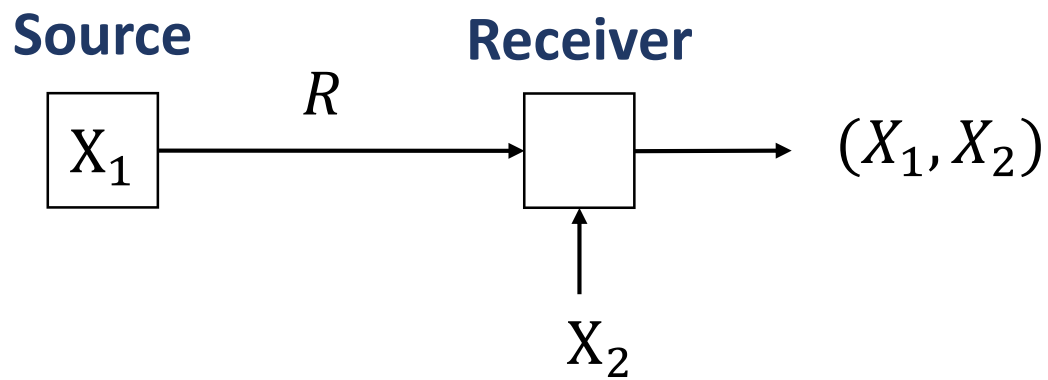
Slepian-Wolf showed that the sum rate for compression is

$$R_1 + R_2 \geq H(X_1, X_2)$$

[Slepian-Wolf, 1973] D. Slepian and J. K. Wolf, "Noiseless coding of correlated information sources," *IEEE Trans. Inf. Theory*, Jul. 1973.

Computing $f(X_1, X_2) = (X_1, X_2)$

Side information (full knowledge of X_2) [Wyner-Ziv, 1976]

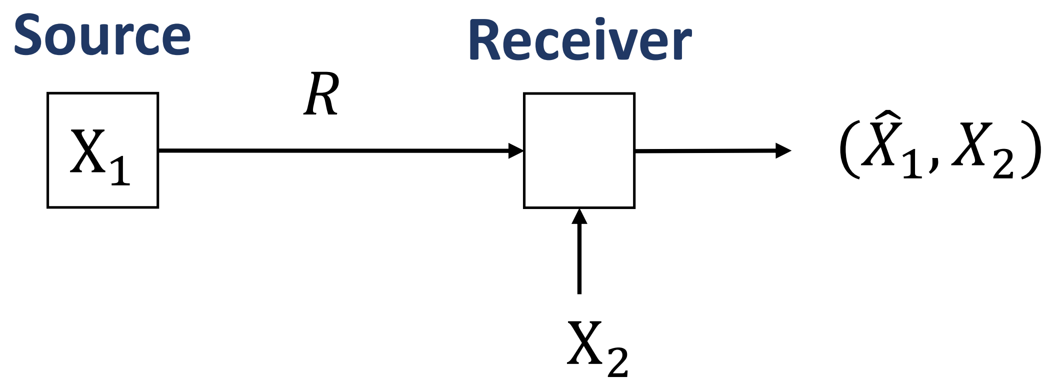


$R \geq H(X_1|X_2)$ under D^*= 0

[Wyner-Ziv, 1976] A. Wyner and J. Ziv, "The rate-distortion function for source coding with side information at the decoder," *IEEE Trans. Inf. Theory*, Jan. 1976.

Computing $f(X_1, X_2) = (X_1, X_2)$

Side information (full knowledge of X_2) [Wyner-Ziv, 1976]



$$R \geq \begin{cases} \inf\limits_{\theta,\beta} \left[\theta\left(h(p_0 * \beta) - h(\beta)\right)\right], & 0 \leq D^* \leq p_0 \\ 0, & D^* \geq p_0 \end{cases}$$

where θ and β satisfy

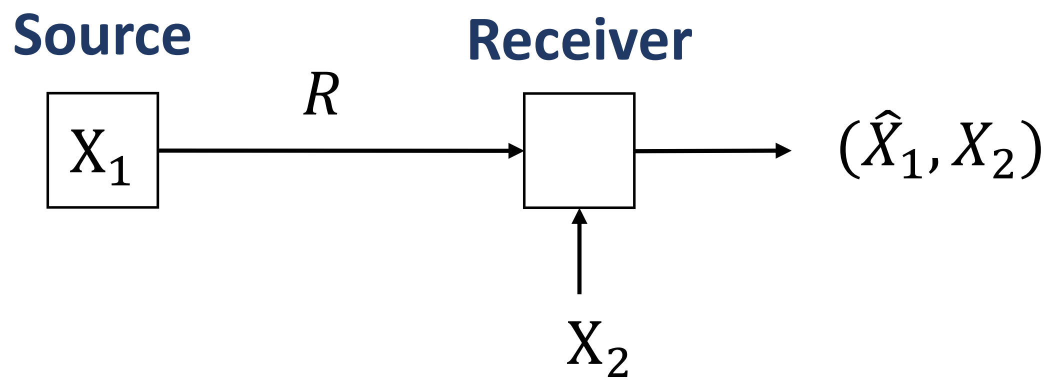
i) $0 \leq \theta \leq 1$, $0 \leq \beta < p_0$

ii) $D^* = \theta\beta + (1-\theta)p_0$ where $p_0 = \min(p, 1-p)$

[Wyner-Ziv, 1976] A. Wyner and J. Ziv, "The rate-distortion function for source coding with side information at the decoder," *IEEE Trans. Inf. Theory*, Jan. 1976.

Computing $f(X_1, X_2) = (X_1, X_2)$

Side information (full knowledge of X_2) [Wyner-Ziv, 1976]

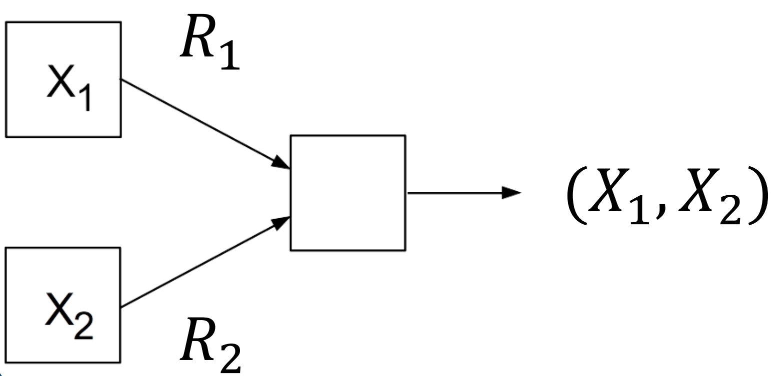


The rate region has a closed-form expression, which can be found by solving an optimization problem

[Wyner-Ziv, 1976] A. Wyner and J. Ziv, "The rate-distortion function for source coding with side information at the decoder," *IEEE Trans. Inf. Theory*, Jan. 1976.

Computing $f(X_1, X_2) = (X_1, X_2)$

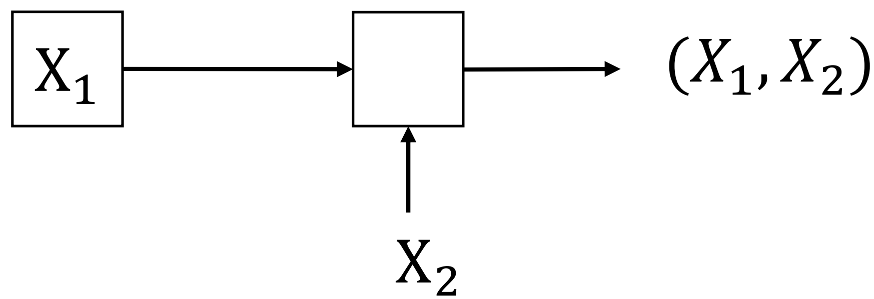
Distributed source compression



[Slepian-Wolf, 1973]

Side information

(full knowledge of X_2)



[Wyner-Ziv, 1976]

[Slepian-Wolf, 1973] D. Slepian and J. K. Wolf, "Noiseless coding of correlated information sources," *IEEE Trans. Inf. Theory*, Jul. 1973.

[Wyner-Ziv, 1976] A. Wyner and J. Ziv, "The rate-distortion function for source coding with side information at the decoder," *IEEE Trans. Inf. Theory*, Jan. 1976.

Achievable Rate Region for Lossless Distributed Communication

The Slepian-Wolf Theorem. The optimal rate region for distributed coding of a 2-DMS $(\mathcal{X}_1 \times \mathcal{X}_2, P_{X_1,X_2})$ is the set of (R_1, R_2) pairs such that [Slepian-Wolf, 1973]

$$R_1 \geq H(X_1|X_2)$$

$$R_2 \geq H(X_2|X_1)$$

$$R_1 + R_2 \geq H(X_1, X_2)$$

[Slepian-Wolf, 1973] D. Slepian and J. K. Wolf, "Noiseless coding of correlated information sources," *IEEE Trans. Inf. Theory*, Jul. 1973.

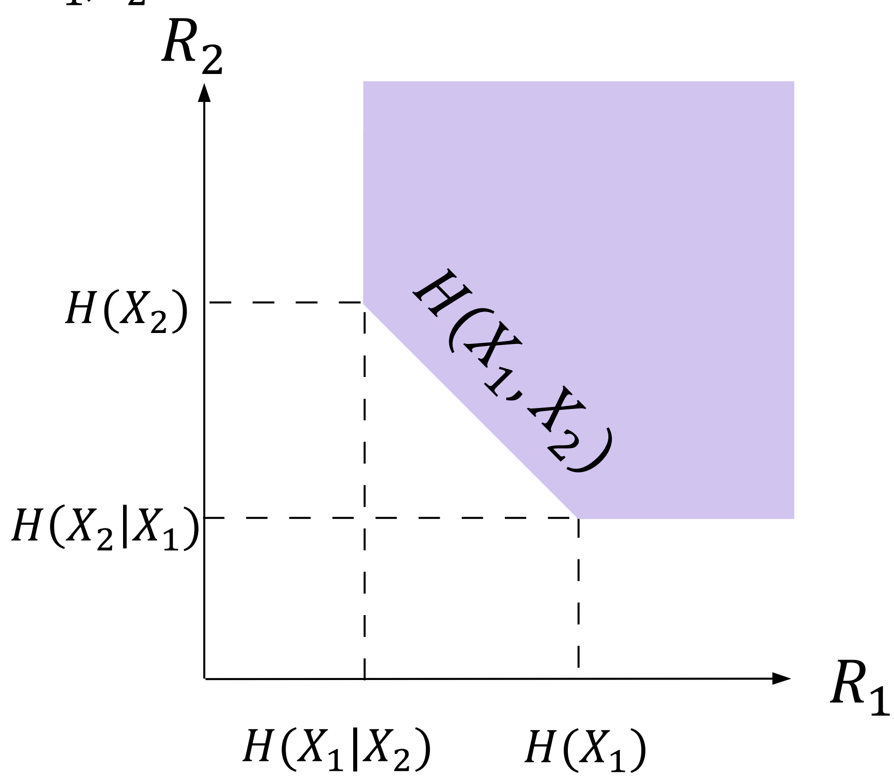
Achievable Rate Region for Lossless Distributed Communication

The Slepian-Wolf Theorem. The optimal rate region for distributed coding of a 2-DMS $(\mathcal{X}_1 \times \mathcal{X}_2, P_{X_1,X_2})$ is the set of (R_1, R_2) pairs such that [Slepian-Wolf, 1973]

$$R_1 \geq H(X_1|X_2)$$

$$R_2 \geq H(X_2|X_1)$$

$$R_1 + R_2 \geq H(X_1, X_2)$$



[Slepian-Wolf, 1973] D. Slepian and J. K. Wolf, "Noiseless coding of correlated information sources," *IEEE Trans. Inf. Theory*, Jul. 1973.

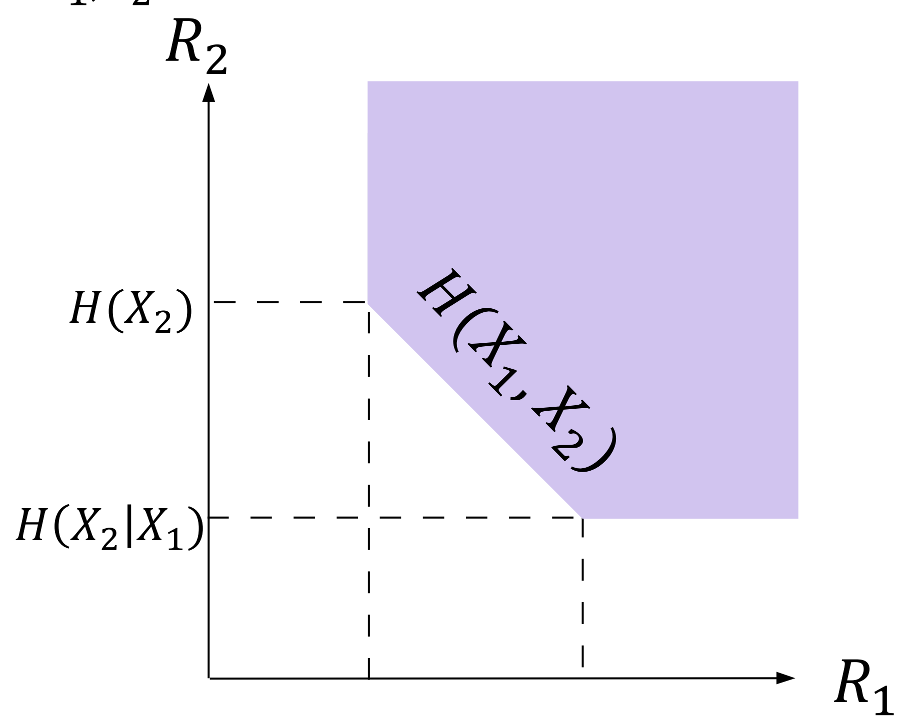
Achievable Rate Region for Lossless Distributed Communication

The Slepian-Wolf Theorem. The optimal rate region for distributed coding of a 2-DMS $(\mathcal{X}_1 \times \mathcal{X}_2, P_{X_1,X_2})$ is the set of (R_1, R_2) pairs such that [Slepian-Wolf, 1973]

$$R_1 \geq H(X_1|X_2)$$

$$R_2 \geq H(X_2|X_1)$$

$$R_1 + R_2 \geq H(X_1, X_2)$$



[Sle

sou

The encoding scheme to achieve these rates relies on orthogonal binning of source sequences.

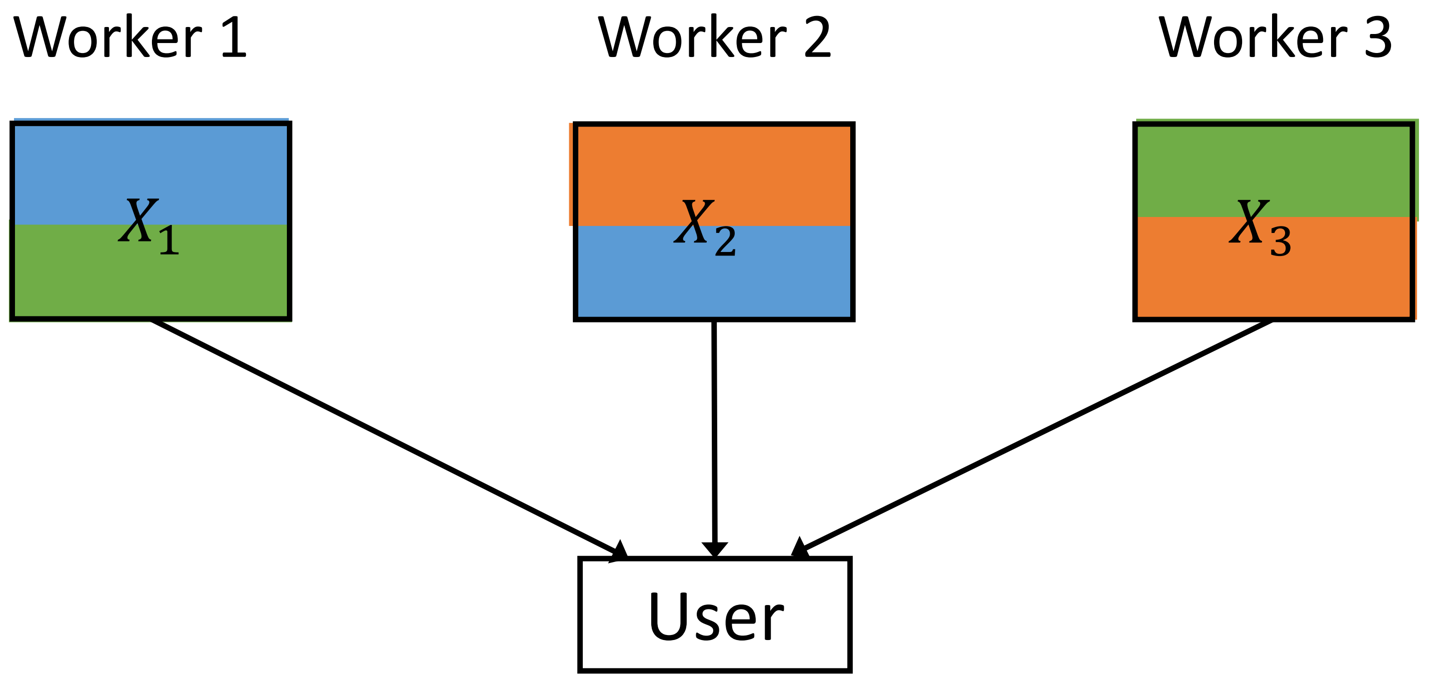
How to compute $f(X_1, X_2) \neq (X_1, X_2)$?

Any ideas?

Distributed functional compression

Distributed Function Computation

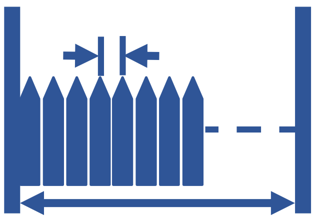
N=3 distributed workers, 1 user node, datasets 



Demand: K_c functions $\{f_j(X_1, X_2, X_3), \ \ j \in [K_c]\}$

Motivation: Why Compress Massive Amount of Data?

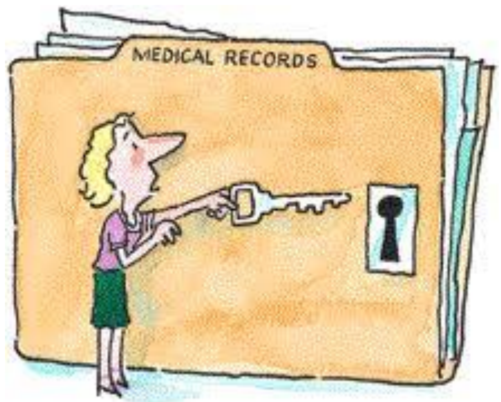
Limited resources & topological constraints

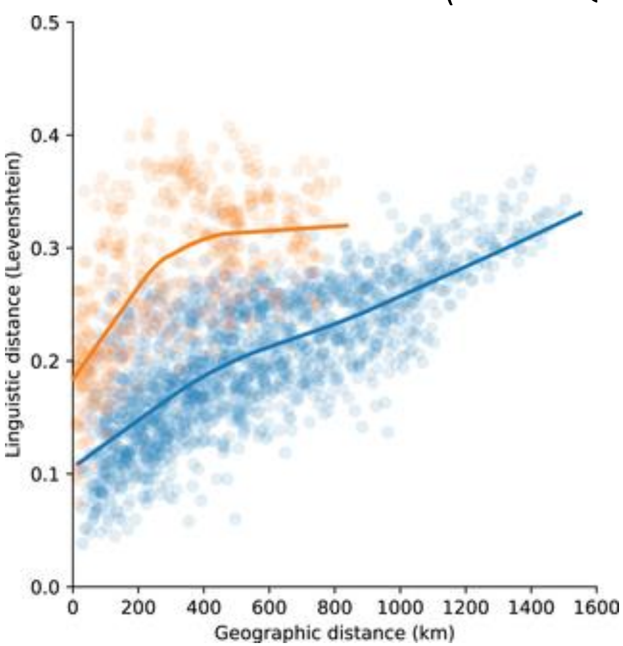


(Source: Qualcomm)

Privacy

sensitive data





Redundancy

geographically dispersed sources

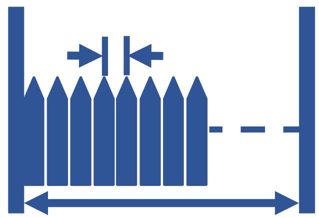
correlation within & across sources

destination only interested in a function of data

Huisman *et al.*,"The geographical configuration of a language area influences linguistic diversity." *PloS one* 14.6, 2019.

Motivation: Why Compress Massive Amount of Data?

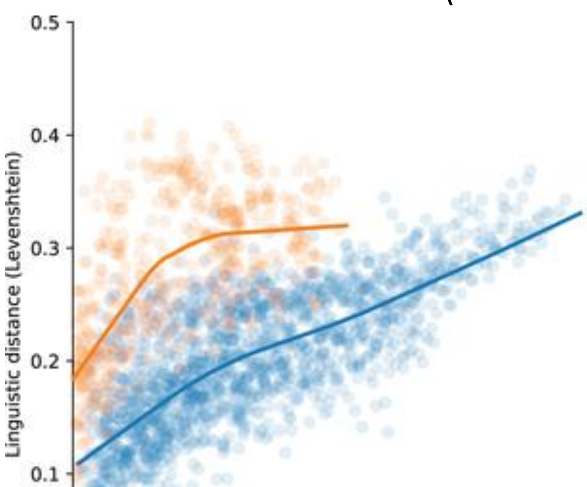
Limited resources & topological constraints



(Source: Qualcomm)

Privacy

sensitive data



Redundancy

geographically dispersed sources

correlation within & across sources

destination only interested in

Goal. to achieve a task (abstracted by a function) rather than transmitting all raw data over a communication network

Huisman *et al.*, "The geographical configuration of a language area influences linguistic diversity." *Plos one* 14.6, 2019.

State of the Art

Distributed source compression [Slepian-Wolf, 1973], [Wyner-Ziv, 1976], practical implementations [Pradhan-Ramchandran, 2013], [Coleman et al, 2006]

Coding for computing, rate region and graph entropy [Körner, 1973], [Alon-Orlitsky, 1996], [Orlitsky-Roche, 2001], [Doshi et al, 2010], [Feizi-Médard, 14], [Feng *et al.*, 2004], [Gallager, 1988], [Kamath-Manjunath, 2008], network flows for computation[Shah *et al.*, 2013], over-the-air computing [Nazer-Gastpar, 2007], [Lim et al., 2019]

Network coding and linear functions [Ho *et al.*, 2006], [Kowshik-Kumar, 2010, 2012], [Appuswamy-Franceschetti, 2014], [Koetter *et al.*, 2004], [Koetter-Médard, 2003], [Huang *et al.*, 2018], [Li *et al.*, 2003]

Parameter estimation [Ozgur, 2018], information theory-based learning [Zheng, Wornell, 2017], principal component analysis [Salamatian *et al.*, 18]

State of the Art

Distributed matrix multiplication [Jia-Jafar, 2021], precision in matrix multiplication [Wang-Jia-Jafar, 2021], secure matrix multiplication [Chang-Tandon, 2018], [Jia-Jafar, 2021], [Chen *et al.*, 2021], [D'Oliveira *et al.*, 2020], matrix multiplication with stragglers [Li *et al.*, 2021]

Coding & communication-computation complexity tradeoffs coded distributed computing [Maddah-Ali-Niesen, 2014], [Dutta *et al.*, 2019], [Li et al, 2016, 2018], [Yu-Maddah-Ali-Avestimehr, 2018], gradient coding [Tandon et al., 2017], communication-computation complexity tradeoffs [Khalesi-Elia, 2022], private information retrieval [Vithana-Banawan-Ulukus, 2021]

Functions with structures [Shen *et al.*, 2018], [Giridhar-Kumar, 2005], [Gorodilova, 2019], linearly separable functions [Wan *et al.*, 2021], nomographic functions [Goldenbaum *et al.*, 2013, 2014], structured codes [Pastore *et al.*, 2023]

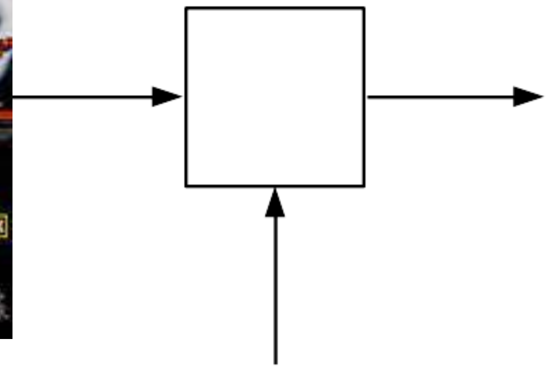
Related work aims to have a joint understanding of structures of networks, functions, and data

Computing $f(X_1, X_2) \neq (X_1, X_2)$

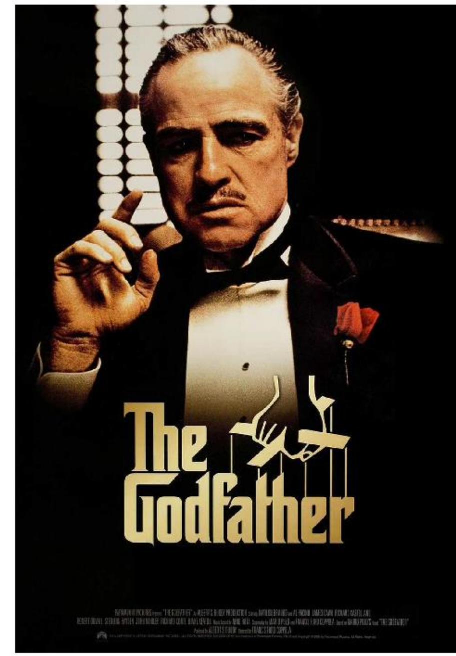
Example: content caching at wireless edge

Source

Receiver



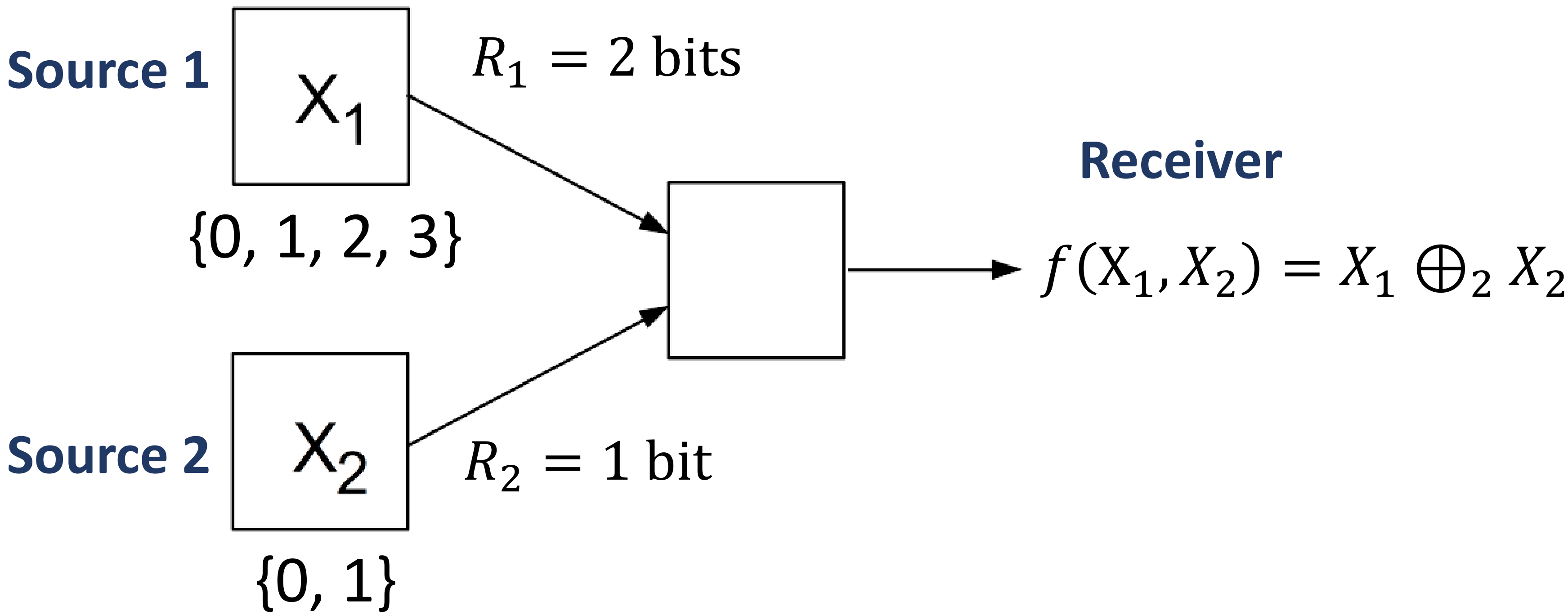
Side information
(#2 IMDb "Top 250")



The receiver is not interested in the entire movie catalog, but a specific function of the catalog.

Computing the Binary XOR Function

Example: binary XOR function [Feizi-Médard, 2014]

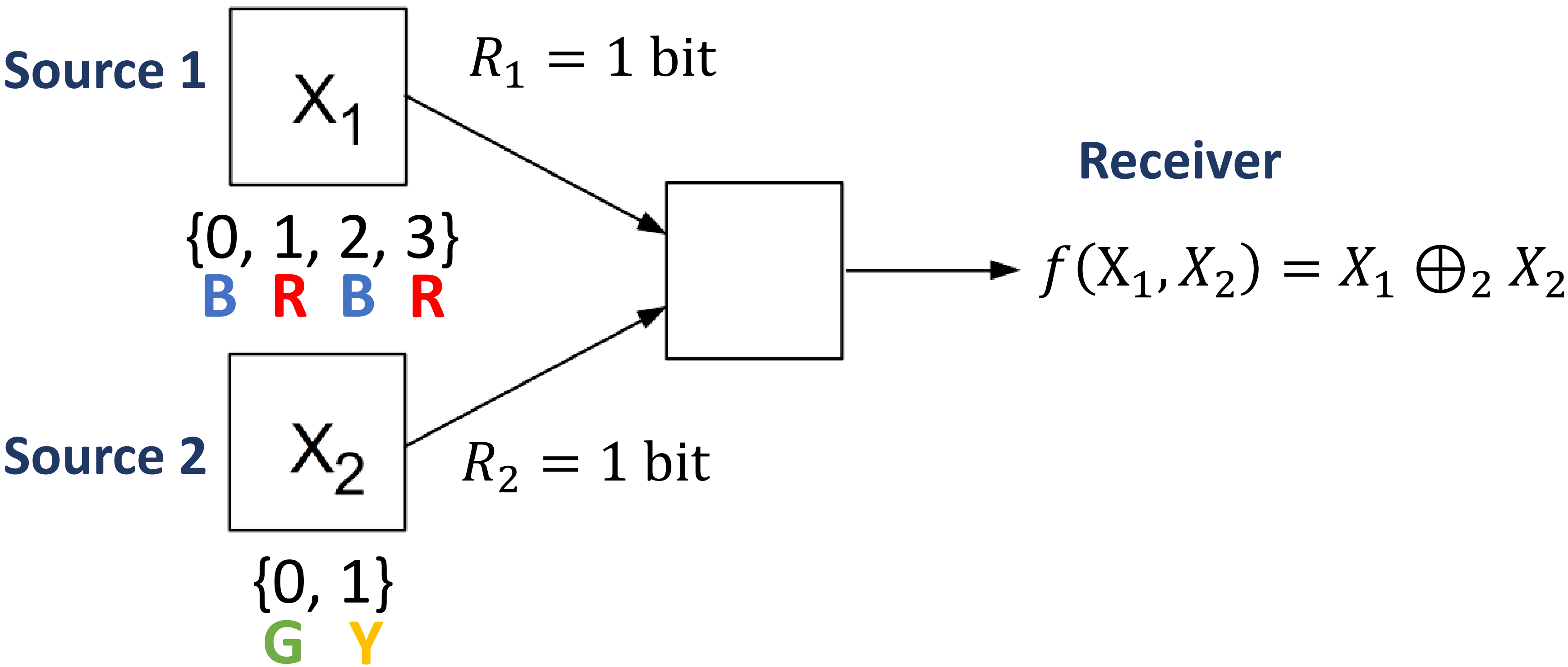


To send the sources in their entirety, we need 3 bits.

[Feizi-Médard, 2014] S. Feizi, M. Médard, "On network functional compression", IEEE Trans. Info. Theory, 2014.

Sending colors instead of sending data

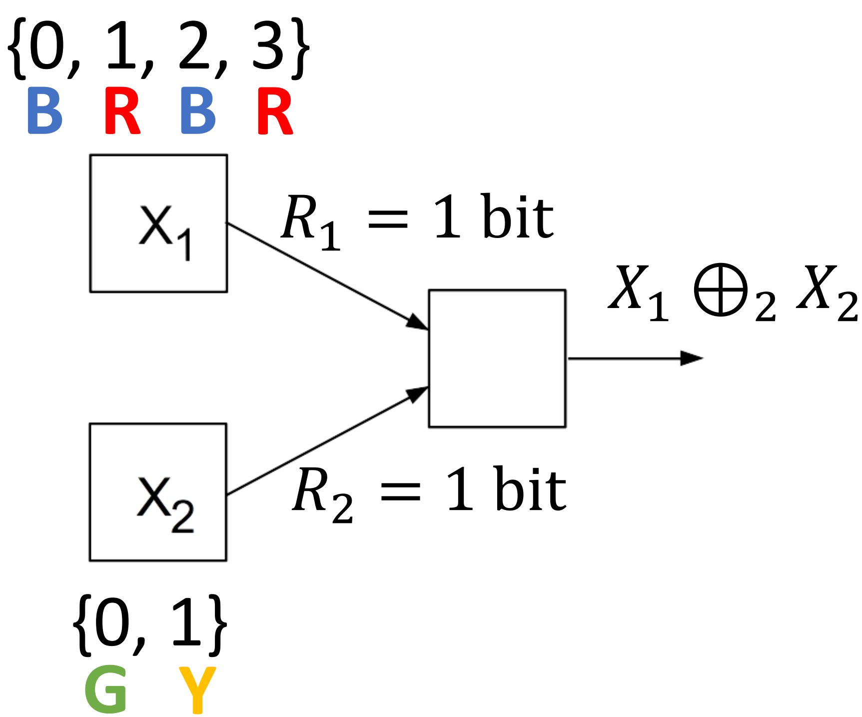
Example: binary XOR function [Feizi-Médard, 2014]

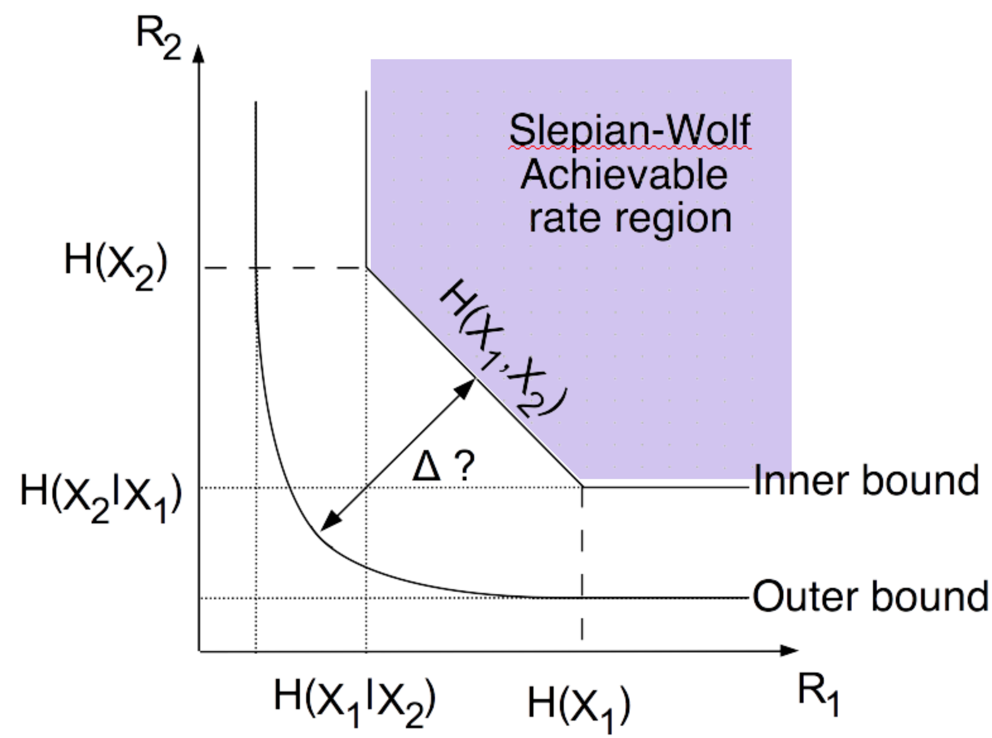


To compute the binary XOR function, we need 2 bits.

Communication Cost for Computing

Example: binary XOR function [Feizi-Médard, 2014]

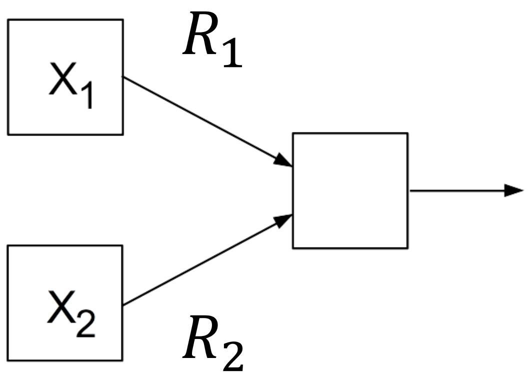




IDEA: Exploit the structure of the function to decide how to distribute computation in networks

Computing the Binary XOR Function

Example: Distributed encoding the modulo two sum of binary sources (DSBS) [Körner-Marton, 1979]



$$f(X_1, X_2) = X_1 \oplus_2 X_2$$

What is the lowest rate pair to reconstruct $f(X_1, X_2)$?

We can use Slepian-Wolf, which requires

$$H(X_1, X_2) = H(X_1\ ) + H(X_2|X_1) = 1 + h(p)$$

Any other guesses?

[Körner, Marton, 1979] J. Körner, K. Marton, “How to encode the modulo two sum of binary sources”, IEEE Trans. Inf. Theory, 1979.

Computing the Binary XOR Function

Example: Distributed encoding the modulo two sum of binary sources (DSBS) [Körner-Marton, 1979]

(i) Choose a binary encoding matrix $A \in \mathbb{F}_2^{k \times n}$ such that

$$\frac{k}{n} \approx H(X_1 \oplus_2 X_2) = h(p)$$

(ii) Source j computes $AX_j^n \in \mathbb{F}_2^{1 \times k}$ and sends

$$AX_j^n$$

(iii) Receiver computes $AX_1^n \oplus_2 AX_2^n = A(X_1^n \oplus_2 X_2^n)$,

(iv) Receiver then recovers $X_1^n \oplus_2 X_2^n$

[Körner, Marton, 1979] J. Körner, K. Marton, “How to encode the modulo two sum of binary sources”, IEEE Trans. Inf. Theory, 1979.

Computing the Binary XOR Function

Example: Distributed encoding the modulo two sum of binary sources (DSBS) [Körner-Marton, 1979]

(i) Choose a binary encoding matrix $A \in \mathbb{F}_2^{k \times n}$ such that

$$\frac{k}{n} \approx H(X_1 \oplus_2 X_2) = h(p)$$

(ii) Source j computes $AX_j^n \in \mathbb{F}_2^{1 \times k}$ and sends

$$AX_j^n$$

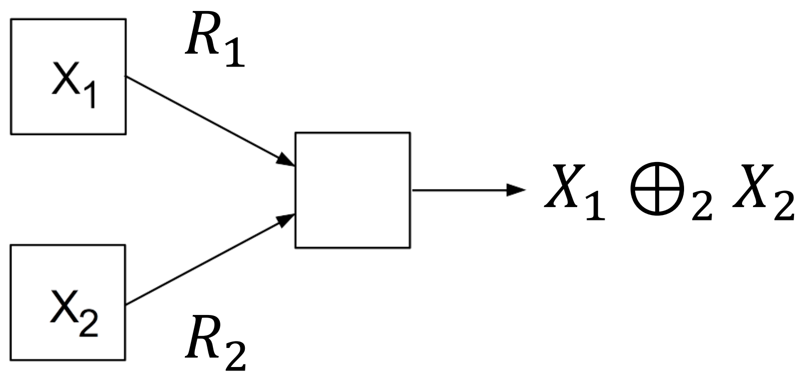
(iii) Receiver computes $AX_1^n \oplus_2 AX_2^n = A(X_1^n \oplus_2 X_2^n)$,

(iv) Receiver then recovers $X_1^n \oplus_2 X_2^n$

[Kö ... ary
sou ...

This scheme is referred to as Structured Binning.

Computing the Binary XOR Function



Sum rate of [Slepian-Wolf, 1973]:

$$H(X_1, X_2) = 1 + h(p)$$

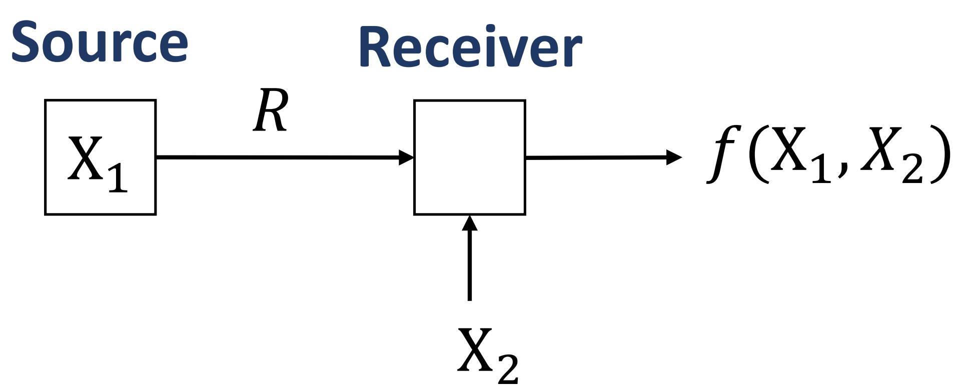
Sum rate of [Körner-Marton, 1979] **(OPTIMAL)**:

$$2H(X_1 \oplus_2 X_2) = 2h(p) \leq 1 + h(p)$$

Körner-Marton approach is constructive, as it captures the structure of functions and sources, but is only valid for the binary XOR function

Computing $f(X_1, X_2) \neq (X_1, X_2)$

Example: Wyner-Ziv type communication system [Yamamoto, 1982]



$$R \geq \inf_{\hat{X}_1 \in \mathcal{P}(\boldsymbol{d})} I(X_1; \hat{X}_1 | X_2)$$

where $\mathcal{P}(\boldsymbol{d})$ is the set of the random variables $\hat{X}_1$ that satisfy

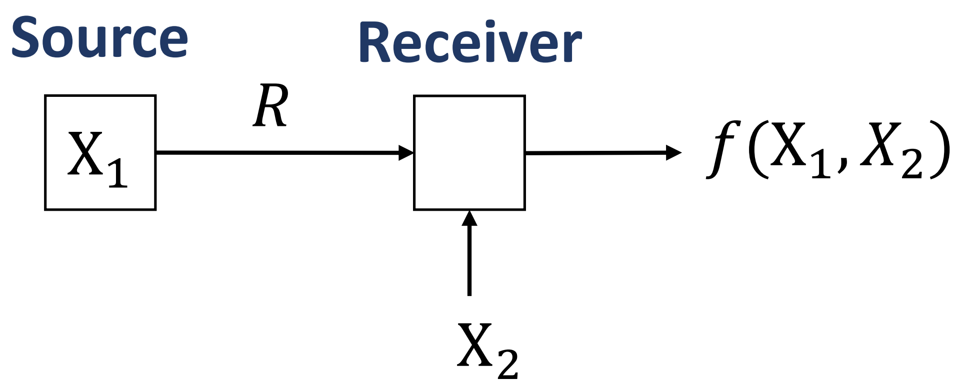
(i) $I\left(X_2; \hat{X}_1 | X_1\right) = 0$ (that is $X_2 - X_1 - \hat{X}_1$ forms a Markov chain)

(ii) $\mathrm{E}\left[D\left(f(X_1, X_2),\ g\left(\hat{X}_1, X_2\right)\right)\right] \leq d$ (average distortion constraint)

[Yamamoto, 1982] H. Yamamoto, "Wyner-Ziv Theory for a General Function of the Correlated Sources", IEEE Trans. Inf. Theory, 1982.

Computing $f(X_1, X_2) \neq (X_1, X_2)$

Example: Wyner-Ziv type communication system [Yamamoto, 1982]



$$R \geq \inf_{\hat{X}_1 \in \mathcal{P}(\boldsymbol{d})} I(X_1; \hat{X}_1 | X_2)$$

whe

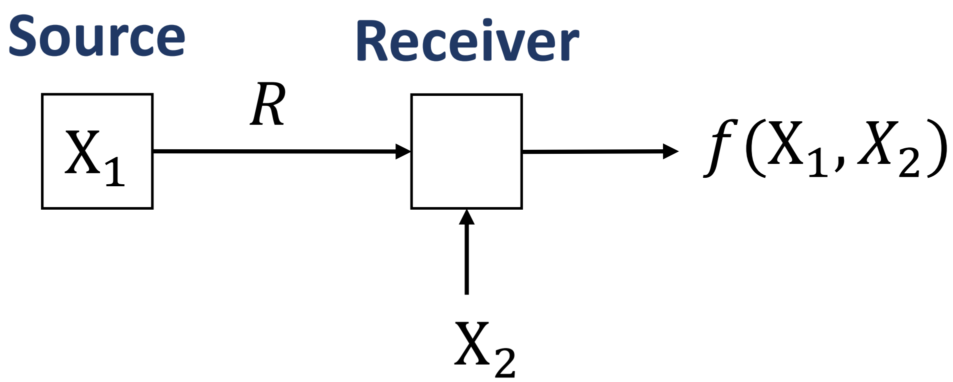
(i) $I($ … in)

Yamamoto does not give a constructive method to encode the source for the specific function.

(ii) $\mathrm{E}\left[D\left(f(X_1, X_2),\ g(X_1, X_2)\right)\right] \leq d$ (average distortion constraint)

[Yamamoto, 1982] H. Yamamoto, "Wyner-Ziv Theory for a General Function of the Correlated Sources", IEEE Trans. Inf. Theory, 1982.

Characteristic Graphs for Computing

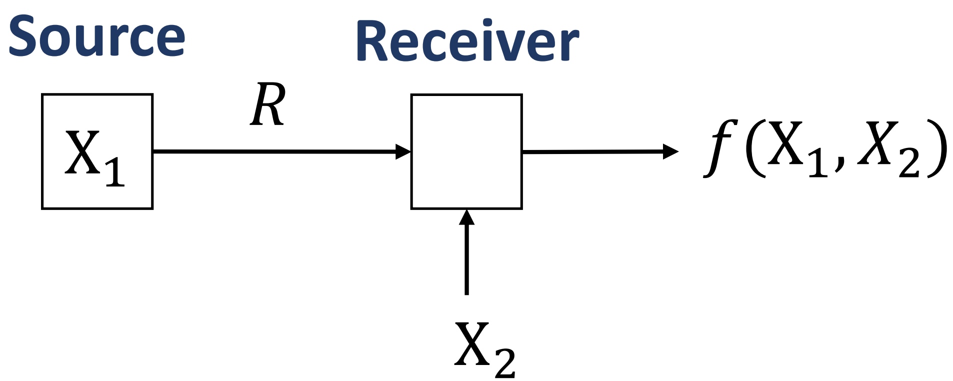


- Recall that **Yamamoto's** lower bound on R [Yamamoto, 1982] **does not give a constructive method to encode** the source for the specific computation task $f(X_1, X_2)$

- The first **constructive approach to functional compression** is devised in [Alon-Orlitsky, 1996].

[Yamamoto, 1982] H. Yamamoto, "Wyner-Ziv Theory for a General Function of the Correlated Sources", IEEE Trans. Inf. Theory, 1982.
[Alon, Orlitsky, 1996] N. Alon and A. Orlitsky, "Source coding and graph entropies," IEEE Trans. Inf. Theory, Sep. 1996.

Characteristic Graphs for Computing



- Recall that **Yamamoto's** lower bound on R [Yamamoto, 1982] **does not give a constructive method to encode** the source for the specific computation task $f(X_1, X_2)$

- The first **constructive approach to functional compression** is devised in [Alon-Orlitsky, 1996].

Alon-Orlitsky gives a constructive graph coloring-based method to encode the source for the specific function.

[Yama
Corre
[Alon
Trans

Characteristic Graphs for Computing

How to build a characteristic graph

$G_{X_1} = (\mathcal{X}_1, \mathcal{E}_1)$ of source X_1?

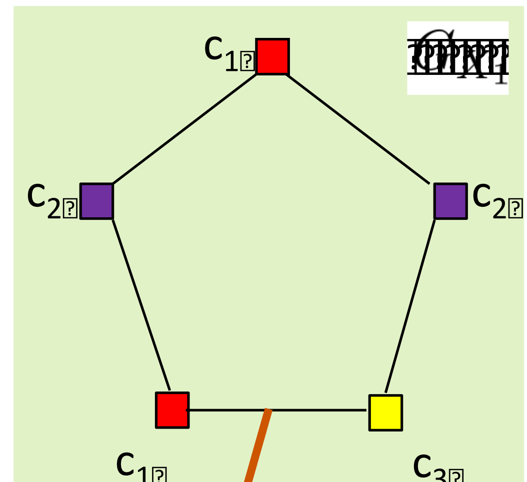
- Vertices are different sample values
- For given vertices $\mathrm{x}_1^1, \mathrm{x}_1^2 \in \mathcal{X}_1$,

  it holds $(\mathrm{x}_1^1, \mathrm{x}_1^2) \in \mathcal{E}_1$ if and only if

  i) $f(\mathrm{x}_1^1, x_2) \neq f(\mathrm{x}_1^2, x_2)$

  ii) $\mathbb{P}(\mathrm{x}_1^1, x_2) \times \mathbb{P}(\mathrm{x}_1^2, x_2) > 0$

Characteristic graph



Two vertices are connected if they should be distinguished.

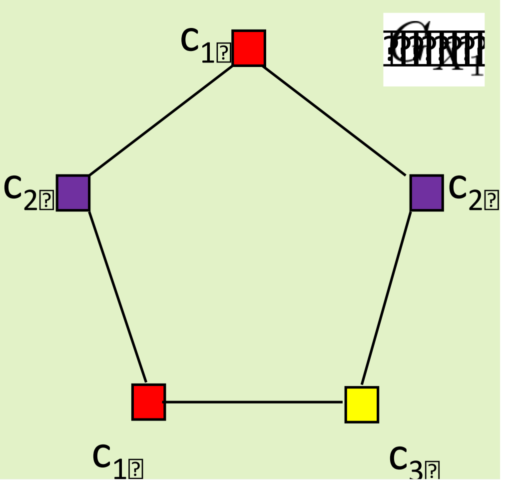
[Alon, Orlitsky, 1996] N. Alon and A. Orlitsky, “Source coding and graph entropies,” IEEE Trans. Inf. Theory, Sep. 1996.

Characteristic Graph and Chromatic Entropy

Example

Let $X_1 \sim \mathrm{Unif}[0:4]$ and X_2 is a random variable such that

Characteristic graph

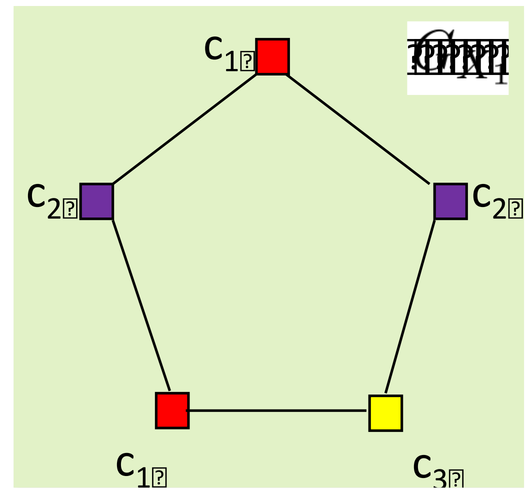


Characteristic Graph and Chromatic Entropy

Example

Let $X_1 \sim \mathrm{Unif}[0:4]$ and X_2 is a random variable such that

Characteristic graph



Assign the set of vertices valid colors:

$$c_{G_{X_1}} = \{c_1, c_2, c_3\}$$

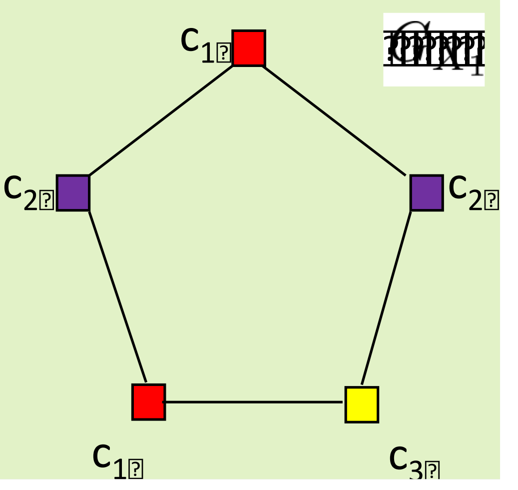
$$P(c_1) = P(c_2) = 2/5, \quad P(c_3) = 1/5$$

Characteristic Graph and Chromatic Entropy

Example

Let $X_1 \sim \mathrm{Unif}[0:4]$ and X_2 is a random variable such that

Characteristic graph



Assign the set of vertices valid colors:

$$c_{G_{X_1}} = \{c_1, c_2, c_3\}$$

$$P(c_1) = P(c_2) = 2/5, \quad P(c_3) = 1/5$$

The chromatic entropy of this graph is

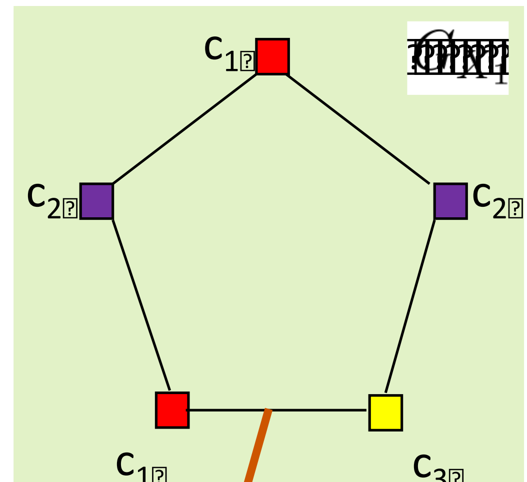
$$H(c_{G_{X_1}}) \approx 1.52 < \mathrm{H}(X_1) = 2.32$$

Characteristic Graph and Chromatic Entropy

Example

Let $X_1 \sim \mathrm{Unif}[0:4]$ and X_2 is a random variable such that

Characteristic graph



Assign the set of vertices valid colors:

$$c_{G_{X_1}} = \{c_1, c_2, c_3\}$$

$$P(c_1) = P(c_2) = 2/5, \quad P(c_3) = 1/5$$

The chromatic entropy of this graph is

$$H(c_{G_{X_1}}) \approx 1.52 < \mathrm{H}(X_1) =$$

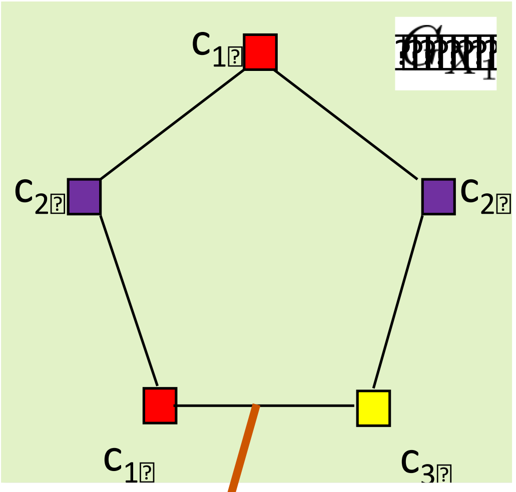
Adjacent vertices have distinct colors.

Characteristic Graph and Chromatic Entropy

Example

Let $X_1 \sim \mathrm{Unif}[0:4]$ and X_2 is a random variable such that

Characteristic graph



Assign the set of vertices valid colors:

$$c_{G_{X_1}} = \{c_1, c_2, c_3\}$$

$$P(c_1) = P(c_2) = 2/5, \quad P(c_3) = 1/5$$

The chromatic entropy of this graph is

$$H(c_{G_{X_1}}) \approx 1.52 < \mathrm{H}(X_1) =$$

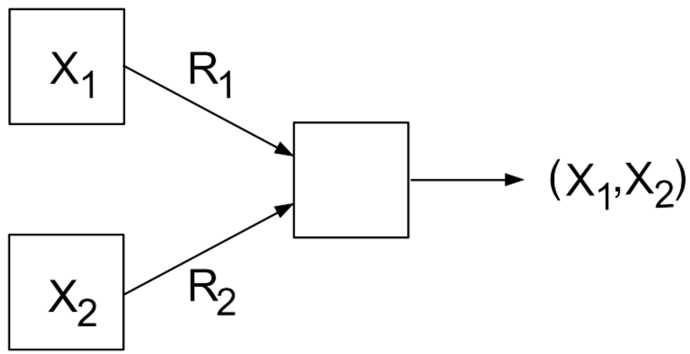
Adjacent vertices have

Exploit Körner's graph entropy [Körner, 1973] to compute the true rate region for distributed functional compression.

[Körner, 1973] J. Körner, "Coding of an information source having ambiguous alphabet and the entropy of graphs," *Prague Conf. Inf.*, 1973.

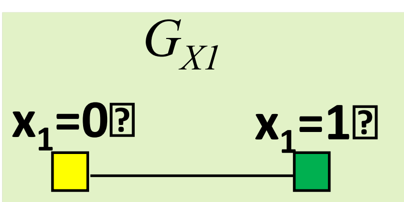
Coloring of Trees is not NP-hard.

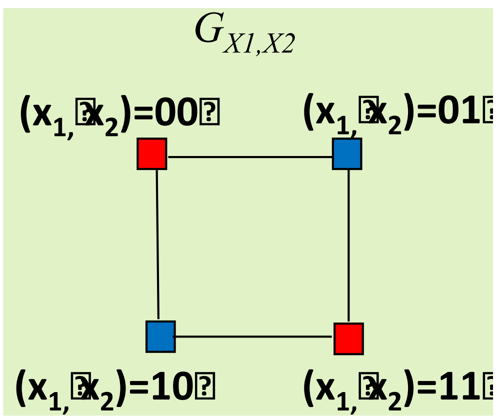
$X_i \sim$ Bern



Sources send their colorings.

Using received colors, the node selects corresponding color from G_{X_1,X_2}





Computations at intermediate nodes reduce the transmission rate!

Körner's Graph Entropy

Chromatic entropy of a graph [Alon-Orlitsky, 1996]

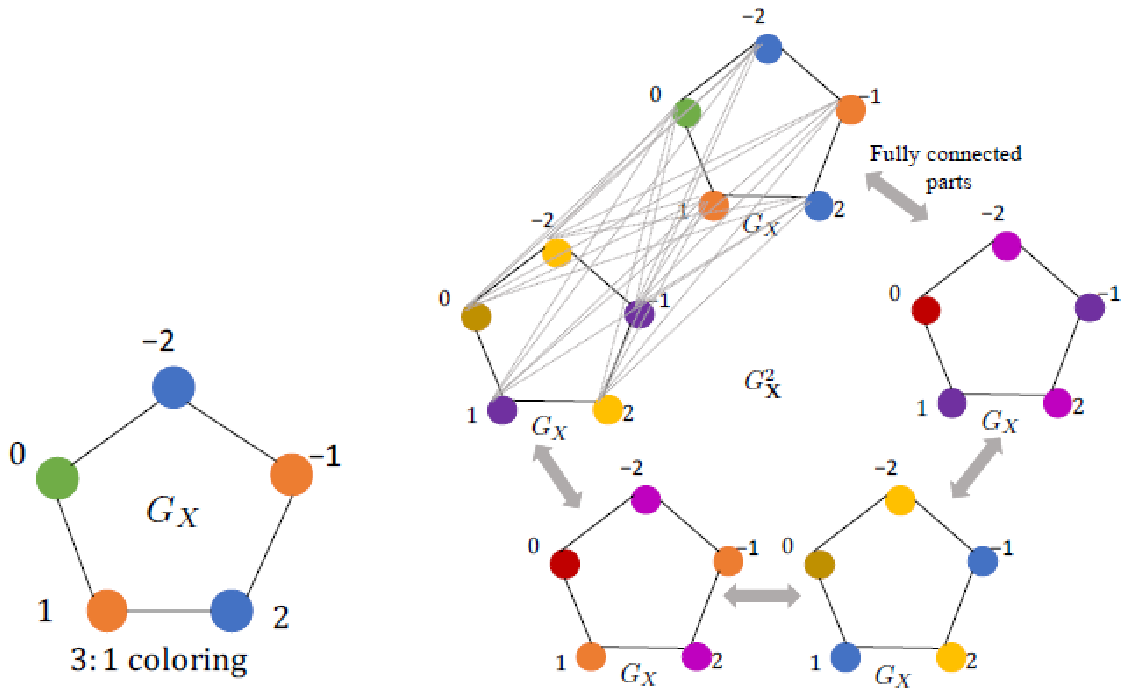
$$H^{\chi}_{G_{X_1}}(X_1) = \min_{c_{G_{X_1}} \text{ is a valid coloring of } G_{X_1}} H(c_{G_{X_1}}(X_1))$$

Körner's theorem for the relation between the chromatic entropy and the graph entropy [Körner, 1973]

$$\lim_{n\to\infty} \frac{1}{n} H^{\chi}_{G^n_{X_1}}(X_1) = H_{G_{X_1}}(X_1)$$

[Alon-Orlitsky, 1996] N. Alon and A. Orlitsky, "Source coding and graph entropies," IEEE Trans. Inf. Theory, Sep. 1996.
[Körner, 1973] J. Korner, "Coding of an information source having ambiguous alphabet and the entropy of graphs," in Proc., Prague Conf. Inf. Theory, 1973.

2nd OR power graph… n-th power graph

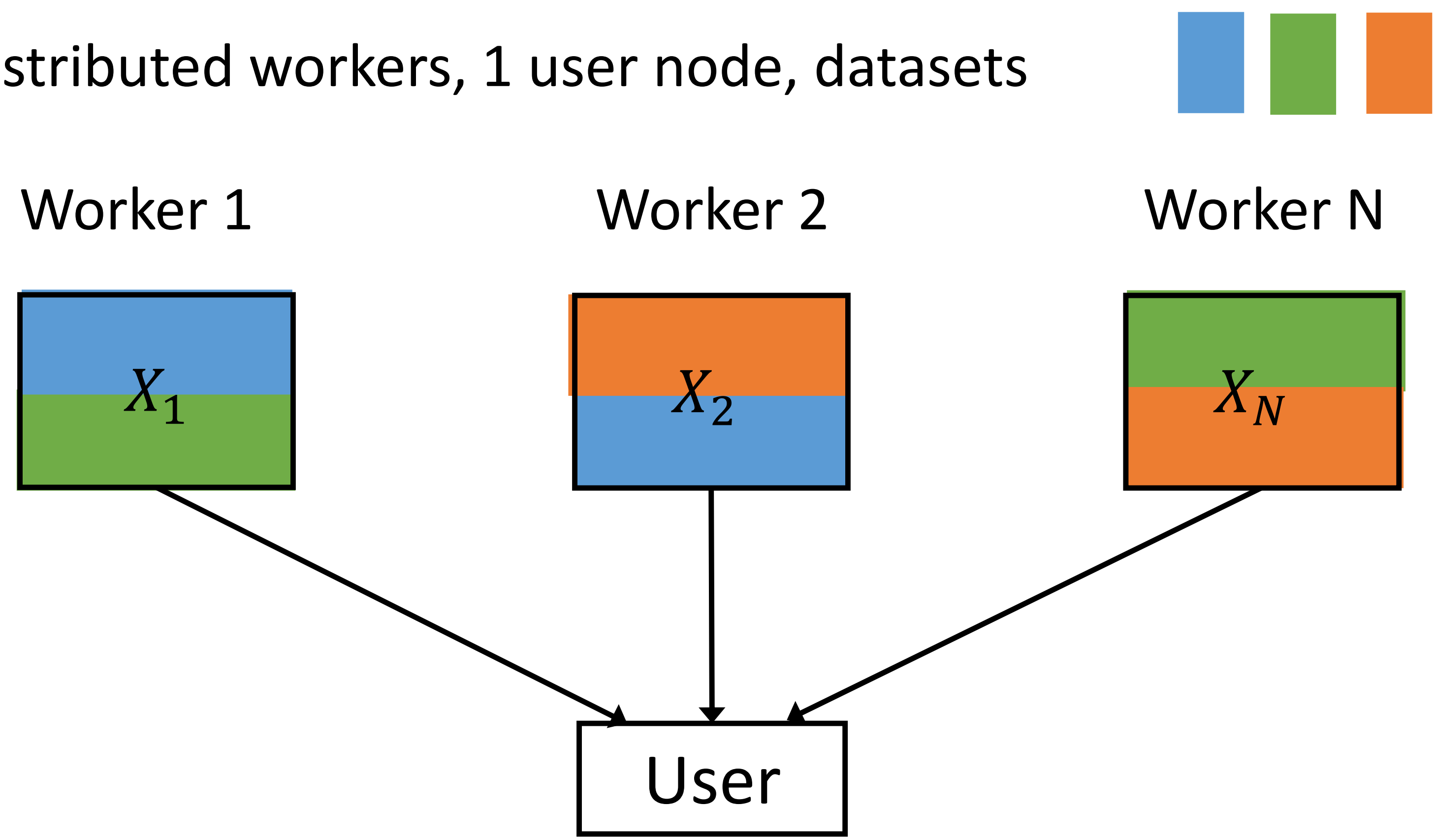


- $G^2_{X_1}$ is the second power graph of the characteristic graph G_{X_1}
- $G^2_{X_1}$ requires 8 colors (vs 9 colors)
- Two subsets of vertices are fully connected if each vertex of one set is connected to every vertex in the other set.

Communication-computation complexity tradeoffs

Distributed Function Computation

N distributed workers, 1 user node, datasets



Demand: K_c functions $\{f_j(X_1, X_2, \ldots, X_N), \;\; j \in [K_c]\}$

General Distributed Computation Setting

K datasets, $W_k,\ k \in [K] = \{1, \dots, K\}$

N distributed workers, $\Omega = [N] = \{1, \dots, N\}$

Datasets assigned to worker $i \in [N]$

$$X_i = \{W_k\}_{k \in Ƶ_i}, \quad \text{where} \quad Ƶ_i \subseteq [K]$$

User wants to recover $K_c \geq 1$ functions $\{f_j(X_1, \dots, X_N), j \in [K_c]\}$

Number of workers user should wait to recover $\{f_j\}$:

$$N_r$$

Minimum computation capacity per worker:

$$M = \frac{K}{N}(N - N_r + 1)$$

Distributed Computation of General Functions

Identity function $f(X_1, \dots, X_N) = (X_1, \dots, X_N)$

Affine function $f(X_1, \dots, X_N) = \sum_{i \in [N]} c_i X_i$

Bilinear functions $f(X_1, \dots, X_N) = \prod_{i \in [N]} c_i X_i$

Matrix multiplication $f(X_1, \dots, X_N) = \boldsymbol{A} \times \boldsymbol{B}$

Sparse polynomials

$$f(X_1, \dots, X_N) = \sum_{j \in [t]:\, \sum_{k \in [K]} l_k \le D} a_{j, l_{[K]}} \prod_{k \in [K]} {W_k}^{l_k}$$

and general nonlinear functions **(multi-shot and one-shot)**

How much rate is needed for computation?
minimum total communication cost of all workers

Computing Matrix Product

N distributed workers and 1 user that wants to recover

$$f(X_1, \ldots, X_N) = \boldsymbol{A} \times \boldsymbol{B} = [\boldsymbol{A}_1 \quad \boldsymbol{A}_2 \quad \ldots \quad \boldsymbol{A}_N] \begin{bmatrix} \boldsymbol{B}_1 \\ \boldsymbol{B}_2 \\ \vdots \\ \boldsymbol{B}_N \end{bmatrix}$$

$$= \boldsymbol{A}_1\boldsymbol{B}_1 + \boldsymbol{A}_2\boldsymbol{B}_2 + \cdots + \boldsymbol{A}_N\boldsymbol{B}_N \in \mathbb{F}^{\boldsymbol{N}\times\boldsymbol{N}}$$

where

$$\boldsymbol{A}_k = \begin{bmatrix} a_{1k} \\ a_{2k} \\ \vdots \\ a_{Nk} \end{bmatrix} \in \mathbb{F}^{\boldsymbol{N}\times\boldsymbol{1}}$$, column k of $\boldsymbol{A} \in \mathbb{F}^{\boldsymbol{N}\times\boldsymbol{N}}$

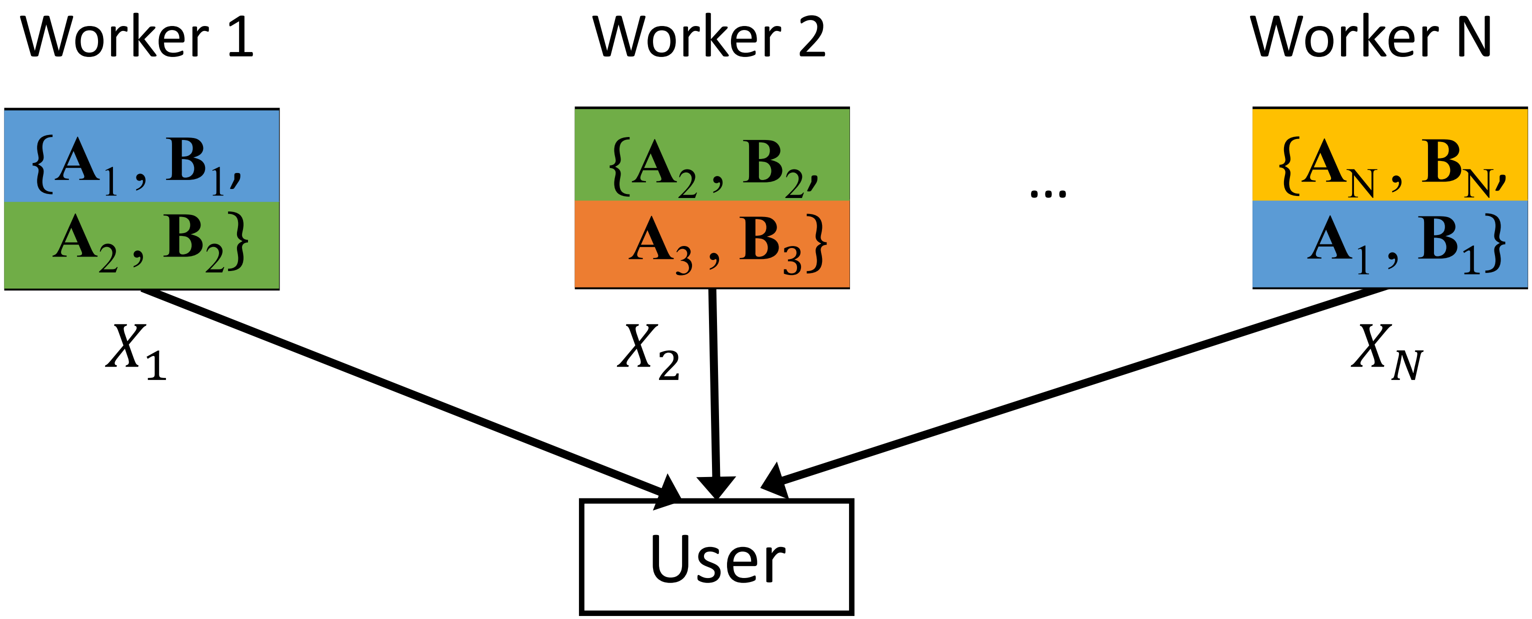
$$\boldsymbol{B}_k = [b_{k1} \quad b_{k2} \quad \ldots \quad b_{kN}] \in \mathbb{F}^{\boldsymbol{1}\times\boldsymbol{N}}$$, row k of $\boldsymbol{B} \in \mathbb{F}^{\boldsymbol{N}\times\boldsymbol{N}}$

Computing Matrix Product

Placement of datasets (subsets of $\boldsymbol{A}_k$ and $\boldsymbol{B}_k$) is cyclic:

$$X_1 = \{W_1, W_2\}, \quad X_2 = \{W_2, W_3\}, \ \dots, \quad X_N = \{W_N, W_1\}$$

where W_k=$\{\boldsymbol{A}_k, \boldsymbol{B}_k\}$, $k \in [N]$.



$$f(X_1, \dots, X_3) = \boldsymbol{A} \times \boldsymbol{B}$$

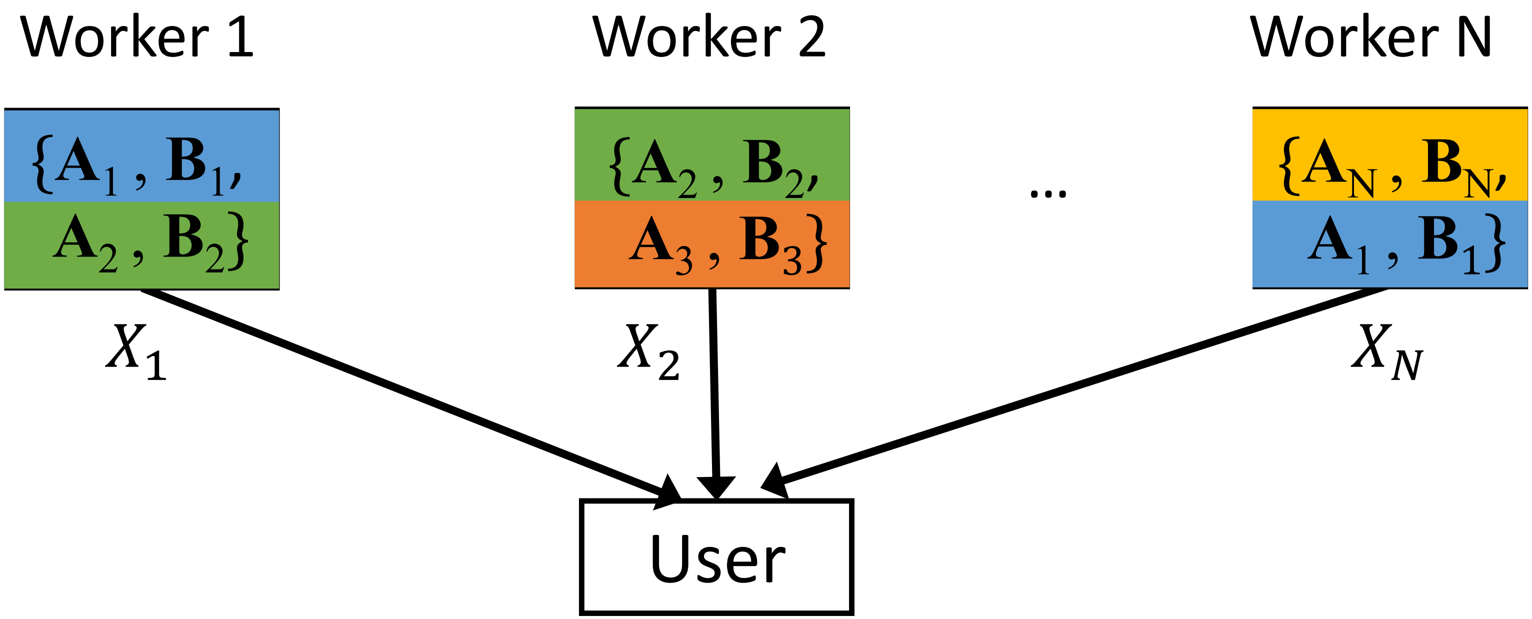
Computing Matrix Product

Number of demanded functions, $K_c = N^2$(an $N \times N$ matrix)

Recovery threshold, $N_r = \mathrm{N} - 1$

Number of datasets, $K = 2N^2$

Cache (or computation) capacity, $M = \frac{K}{N}(N - N_r + 1) = 4N$



$$f(X_1, \dots, X_N) = \boldsymbol{A} \times \boldsymbol{B}$$

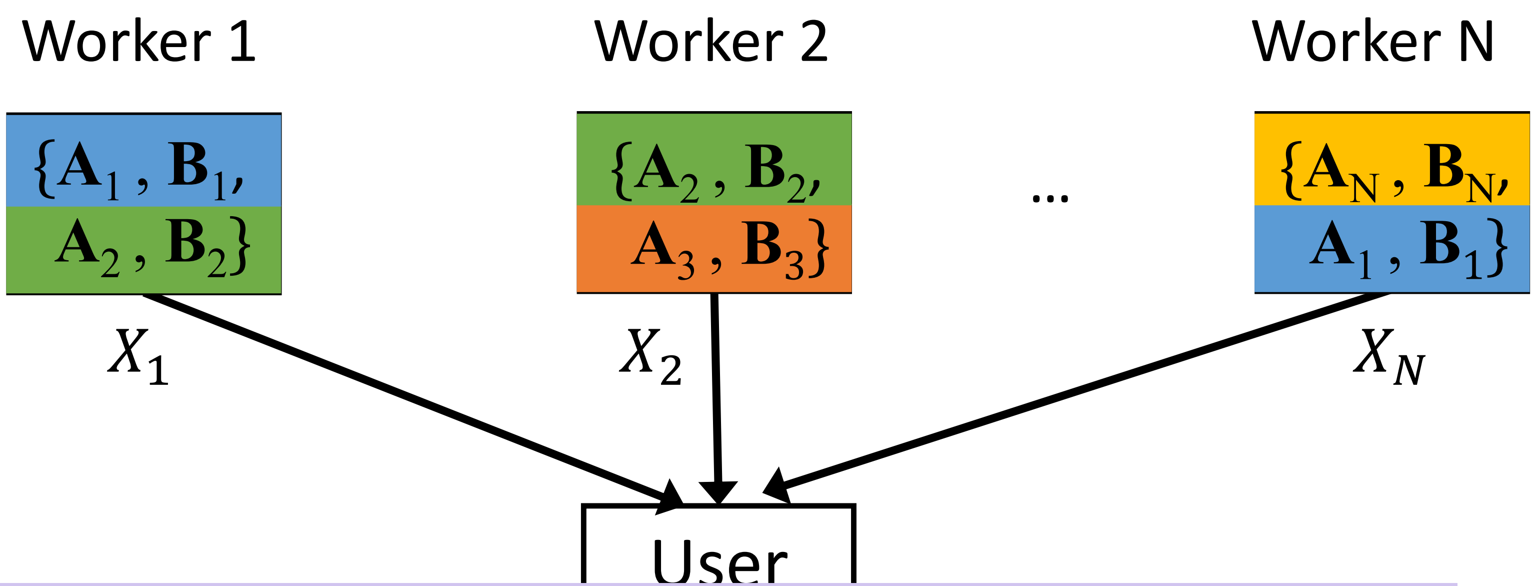
Computing Matrix Product

Number of demanded functions, $K_c = N^2$ (an $N \times N$ matrix)

Recovery threshold, $N_r = \mathrm{N} - 1$

Number of datasets, $K = 2N^2$

Cache (or computation) capacity, $M = \frac{K}{N}(N - N_r + 1) = 4N$

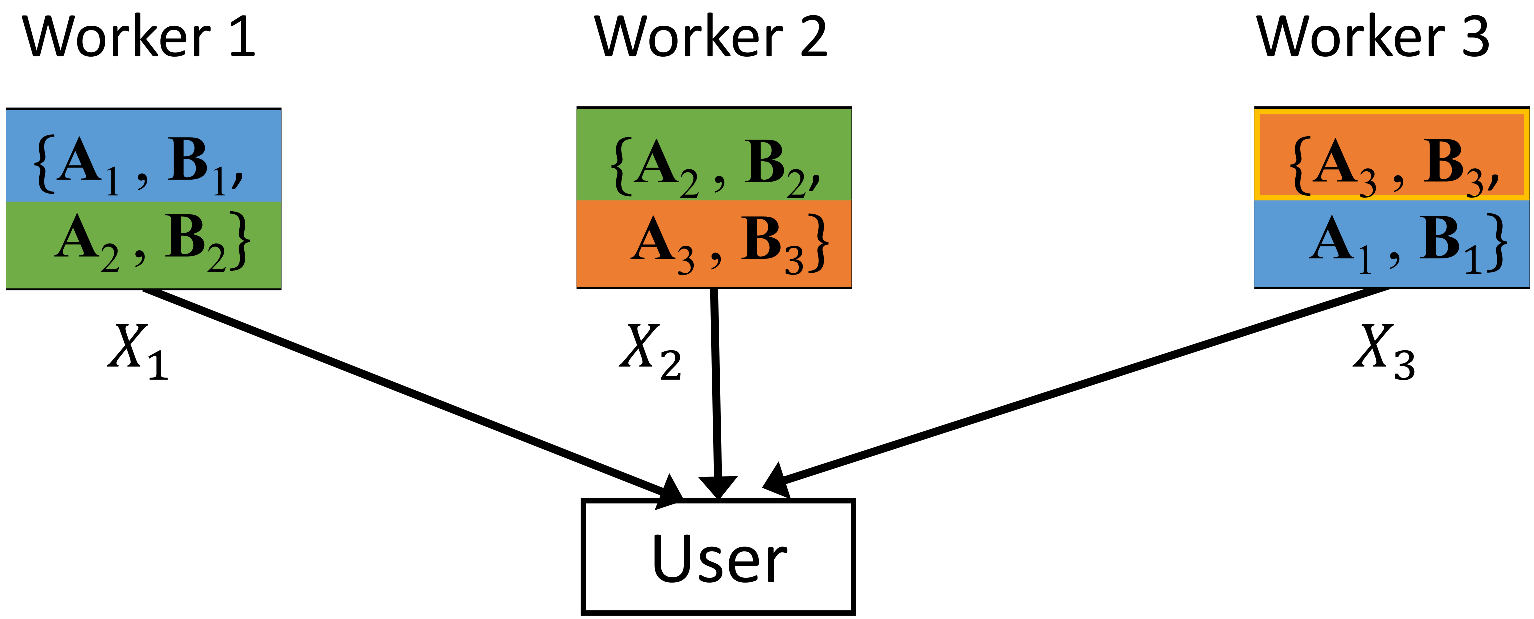


How much rate is needed for computation?

minimum total communication cost of all workers

Example: Computing Matrix Product

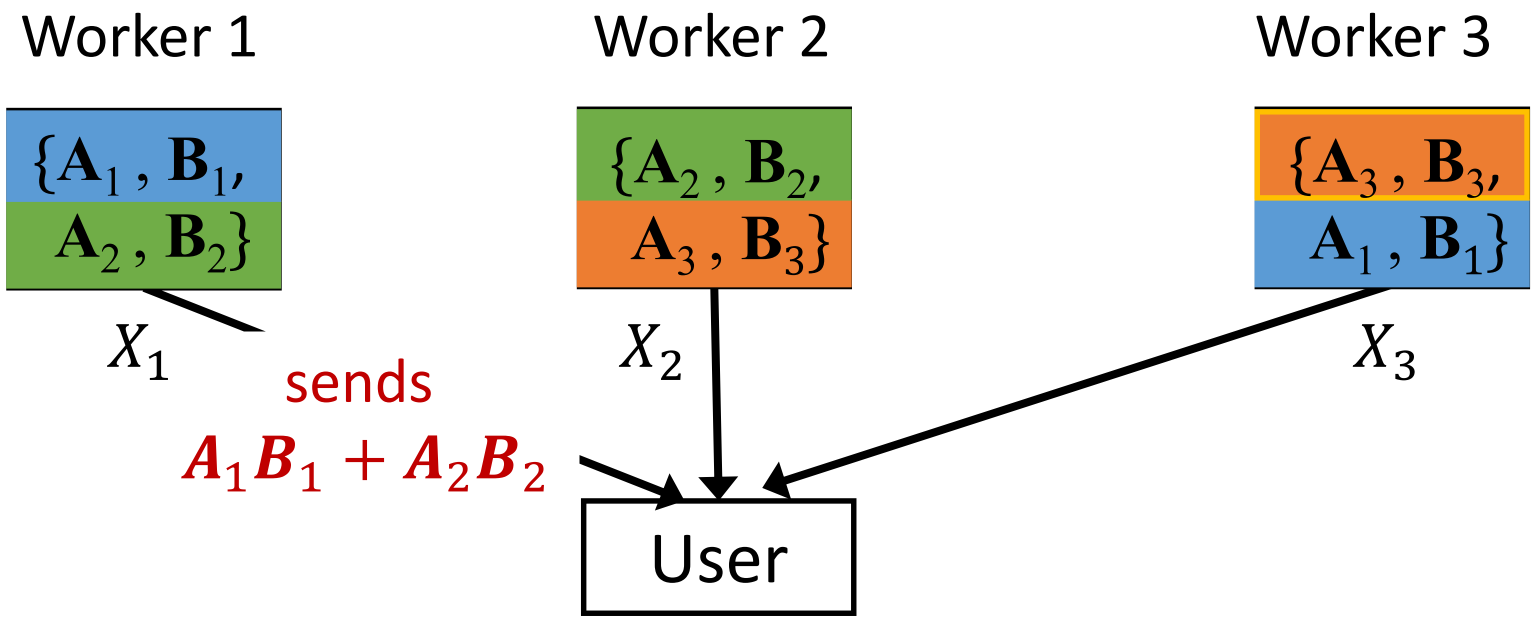
Number of demanded functions, $K_c = 9$ (an 3×3 matrix)
Recovery threshold, $N_r = 2$
Number of datasets, $K = 18$
Cache (or computation) capacity, $M = 12$



$$f(X_1, \ldots, X_3) = \boldsymbol{A} \times \boldsymbol{B} = \boldsymbol{A}_1\boldsymbol{B}_1 + \boldsymbol{A}_2\boldsymbol{B}_2 + \boldsymbol{A}_3\boldsymbol{B}_3$$

Example: Computing Matrix Product

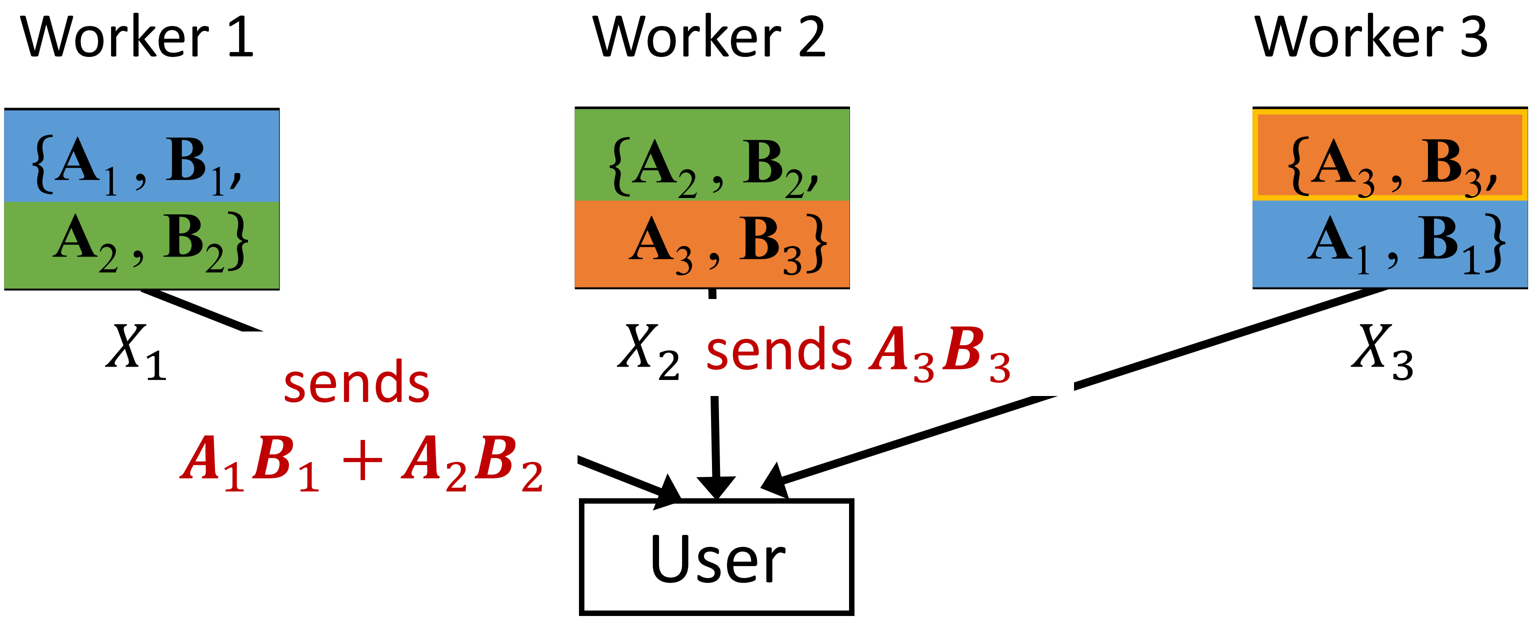
Number of demanded functions, $K_c = 9$ (an 3×3 matrix)
Recovery threshold, $N_r = 2$
Number of datasets, $K = 18$
Cache (or computation) capacity, $M = 12$



$$f(X_1, \dots, X_3) = \boldsymbol{A} \times \boldsymbol{B} = \boldsymbol{A}_1\boldsymbol{B}_1 + \boldsymbol{A}_2\boldsymbol{B}_2 + \boldsymbol{A}_3\boldsymbol{B}_3$$

Example: Computing Matrix Product

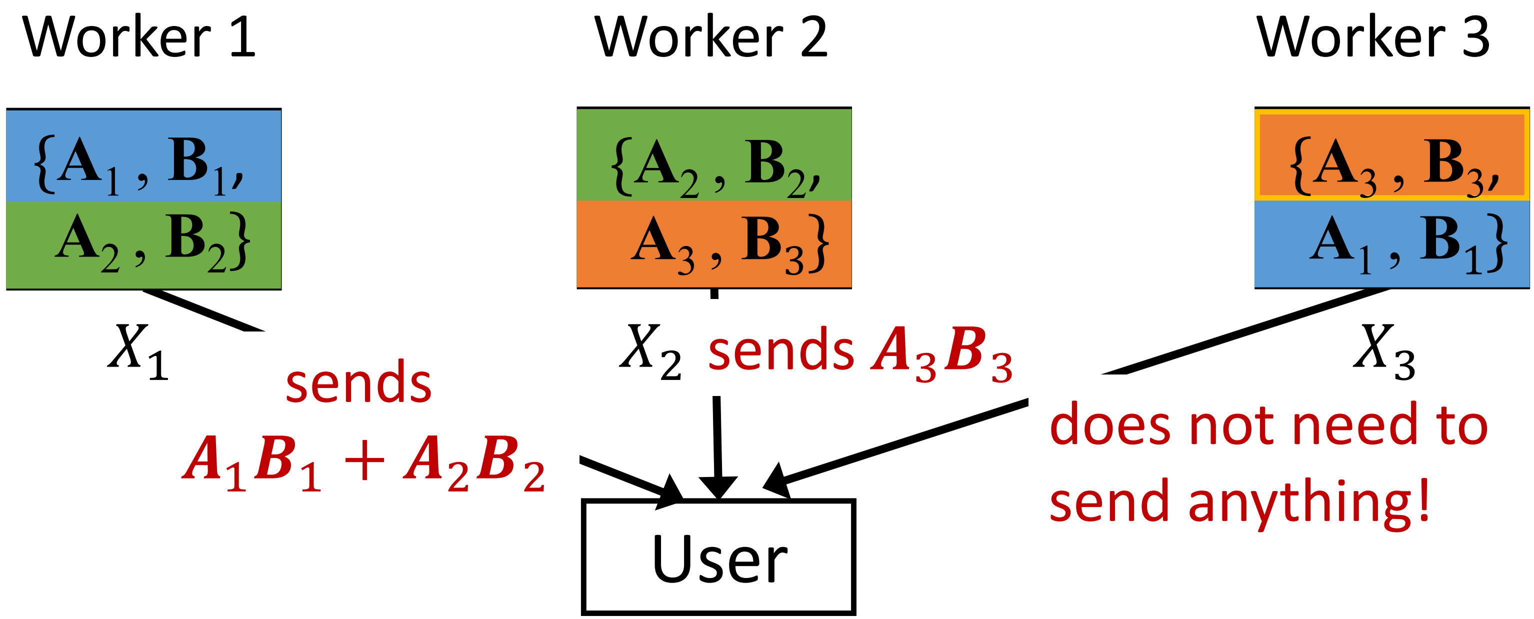
Number of demanded functions, $K_c = 9$ (an 3× 3 matrix)
Recovery threshold, $N_r = 2$
Number of datasets, $K = 18$
Cache (or computation) capacity, $M = 12$



$$f(X_1, \ldots, X_3) = \boldsymbol{A} \times \boldsymbol{B} = \boldsymbol{A}_1\boldsymbol{B}_1 + \boldsymbol{A}_2\boldsymbol{B}_2 + \boldsymbol{A}_3\boldsymbol{B}_3$$

Example: Computing Matrix Product

Number of demanded functions, $K_c = 9$ (an $3\times$ 3 matrix)
Recovery threshold, $N_r = 2$
Number of datasets, $K = 18$
Cache (or computation) capacity, $M = 12$



$$f(X_1, \dots, X_3) = \boldsymbol{A} \times \boldsymbol{B} = \boldsymbol{A}_1\boldsymbol{B}_1 + \boldsymbol{A}_2\boldsymbol{B}_2 + \boldsymbol{A}_3\boldsymbol{B}_3$$

Summary of the Lecture

- Distributed computation

  motivation, challenges

- Distributed source compression for computation
- Existing results, examples for the asymptotic rates for

  special function classes (+, x, A×B,…)

- Distributed computation in broader topologies, functions of correlated data

  Exploit structures in data, function, and topology

Thank You!

Questions?

derya.malak@eurecom.fr