Coded Caching for Content Distribution

Urs Niesen

MobiHoc 2018

Importance of Content Distribution

- Video on demand is driving network traffic growth
  - Netflix streaming service, Amazon Prime Video, Hulu, Verizon / Comcast on Demand, ...
- IP video traffic is predicted to make up 82% of all IP traffic by 2021[1]

[1] Cisco, "The Zettabyte era: Trends and analysis," Tech. Rep., Jun. 2017.

Importance of Content Distribution

- Video on demand is driving network traffic growth
  - Netflix streaming service, Amazon Prime Video, Hulu, Verizon / Comcast on Demand, ...
- IP video traffic is predicted to make up 82% of all IP traffic by 2021[1]
- Places significant stress on service provider's networks

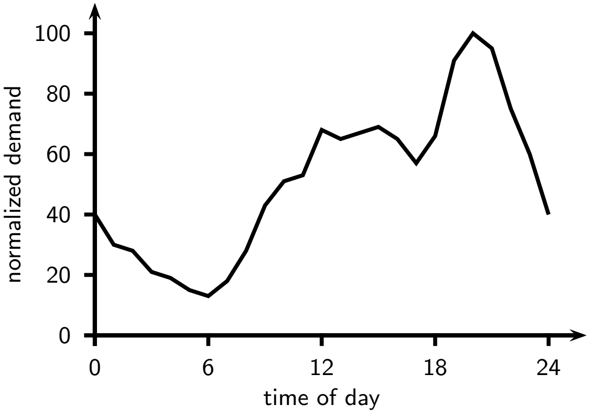
[1] Cisco, "The Zettabyte era: Trends and analysis," Tech. Rep., Jun. 2017.

Importance of Content Distribution

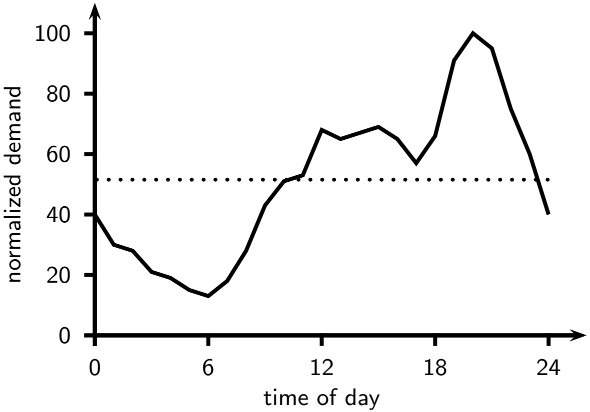
- Video on demand is driving network traffic growth
  - Netflix streaming service, Amazon Prime Video, Hulu, Verizon / Comcast on Demand, ...
- IP video traffic is predicted to make up 82% of all IP traffic by 2021[1]
- Places significant stress on service provider's networks
- Caching (prefetching) can be used to mitigate this stress

[1] Cisco, "The Zettabyte era: Trends and analysis," Tech. Rep., Jun. 2017.

Caching (Prefetching)

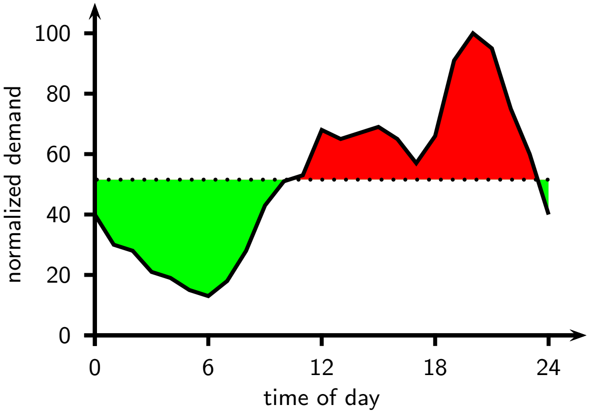


Caching (Prefetching)



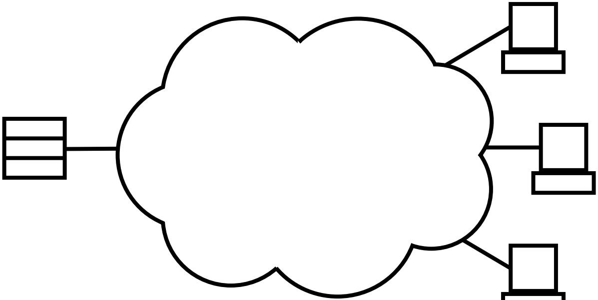
- High temporal traffic variability

Caching (Prefetching)

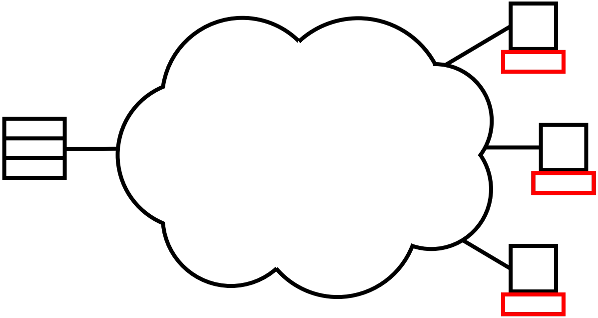


- High temporal traffic variability
- Caching can help smooth traffic

Caching (Prefetching)

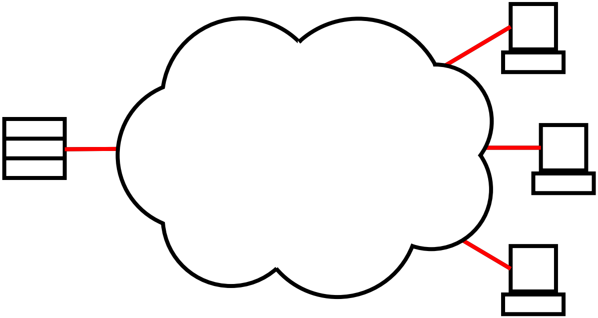


Caching (Prefetching)



- Placement phase (5am): Populate caches

Caching (Prefetching)



- Placement phase (5am): Populate caches
- Delivery phase (8pm): Request and deliver movies

The Role of Caching

Conventional beliefs about caching:

The Role of Caching

Conventional beliefs about caching:

- Caches useful to deliver content locally

The Role of Caching

Conventional beliefs about caching:

- Caches useful to deliver content locally
- Local cache size matters

The Role of Caching

Conventional beliefs about caching:

- Caches useful to deliver content locally
- Local cache size matters
- Statistically identical users $\Rightarrow$ identical cache content

The Role of Caching

Conventional beliefs about caching:

- Caches useful to deliver content locally
- Local cache size matters
- Statistically identical users $\Rightarrow$ identical cache content

This talk will argue:

The Role of Caching

Conventional beliefs about caching:

- ~~Caches useful to deliver content locally~~
- Local cache size matters
- Statistically identical users $\Rightarrow$ identical cache content

This talk will argue:

- The main gain in caching is global

The Role of Caching

Conventional beliefs about caching:

- ~~Caches useful to deliver content locally~~
- ~~Local cache size matters~~
- Statistically identical users ⇒ identical cache content

This talk will argue:

- The main gain in caching is global
- Global cache size matters

The Role of Caching

Conventional beliefs about caching:

- ~~Caches useful to deliver content locally~~
- ~~Local cache size matters~~
- ~~Statistically identical users $\Rightarrow$ identical cache content~~

This talk will argue:

- The main gain in caching is global
- Global cache size matters
- Statistically identical users $\Rightarrow$ different cache content

The Role of Caching

Conventional beliefs about caching:

- ~~Caches useful to deliver content locally~~
- ~~Local cache size matters~~
- ~~Statistically identical users $\Rightarrow$ identical cache content~~

This talk will argue:

- The main gain in caching is global
- Global cache size matters
- Statistically identical users $\Rightarrow$ different cache content
- Coded multicasting as key enabler

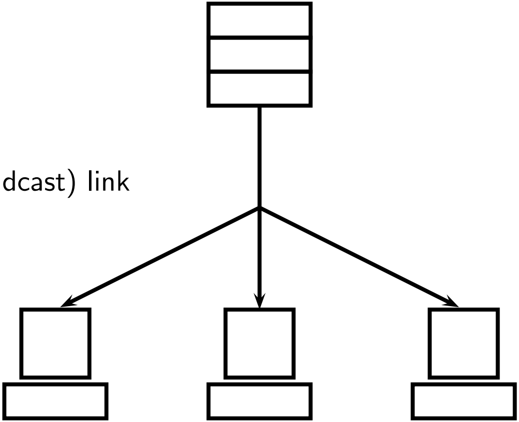
Problem Setting

server

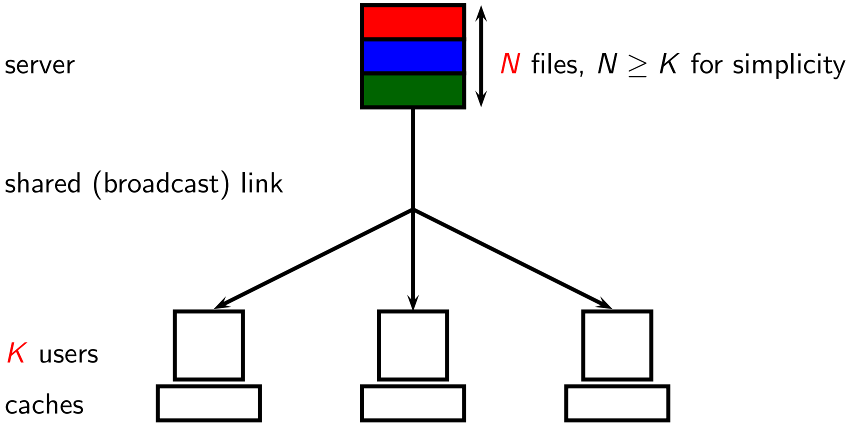
shared (broadcast) link

K users

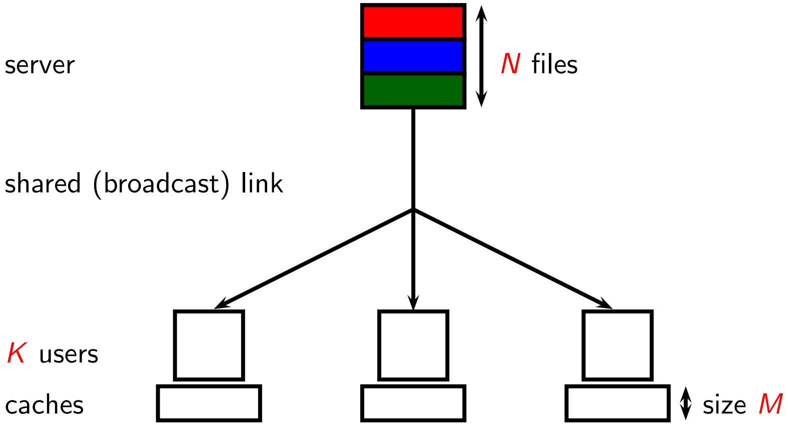
caches



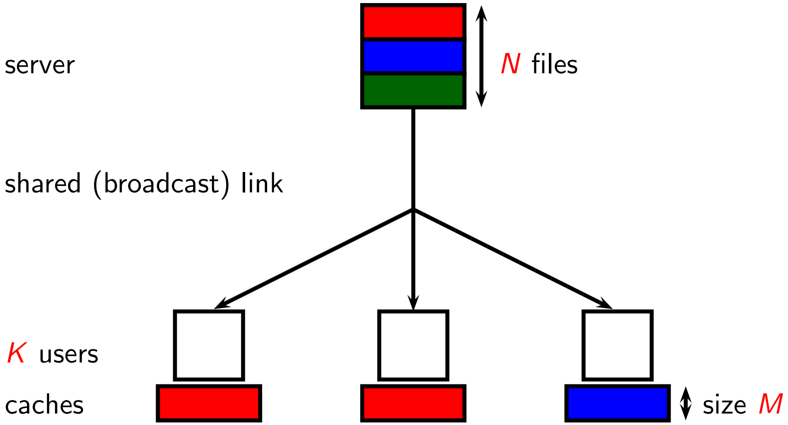
Problem Setting



Problem Setting

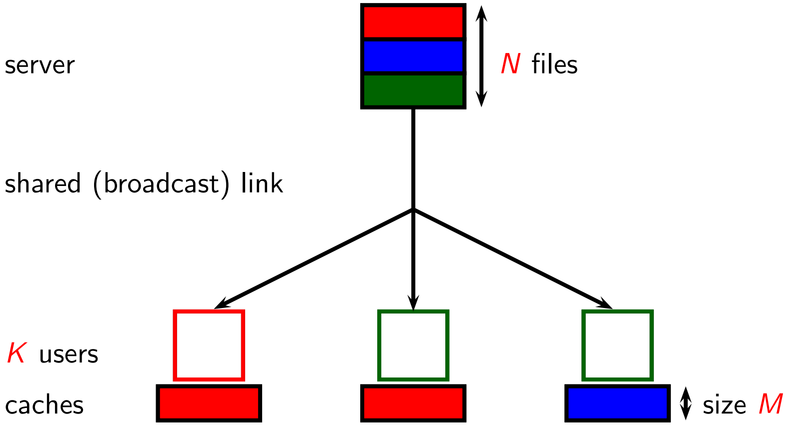


Problem Setting



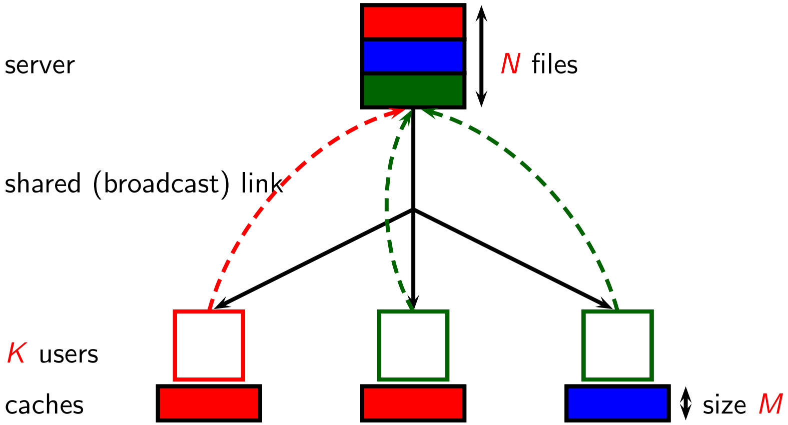
Placement: cache arbitrary function of files (linear, nonlinear, ...)

Problem Setting



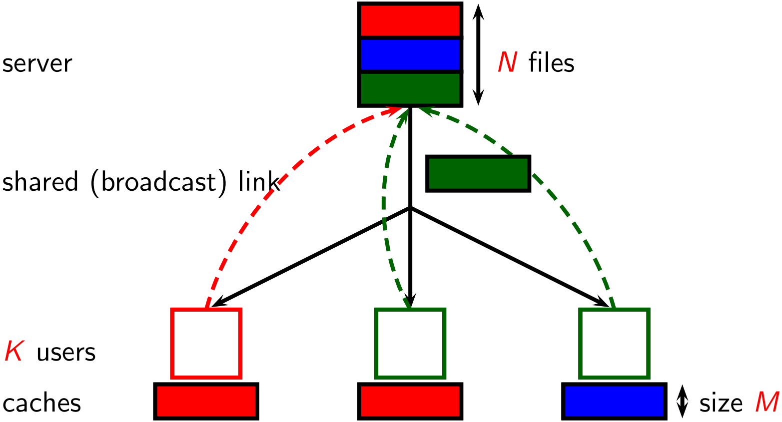
Delivery:

Problem Setting



Delivery: - requests are revealed to server

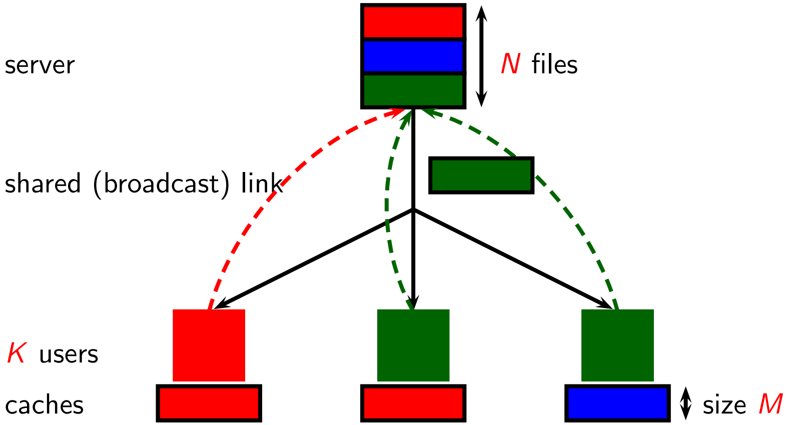
Problem Setting



Delivery: - requests are revealed to server

- server sends arbitrary function of files

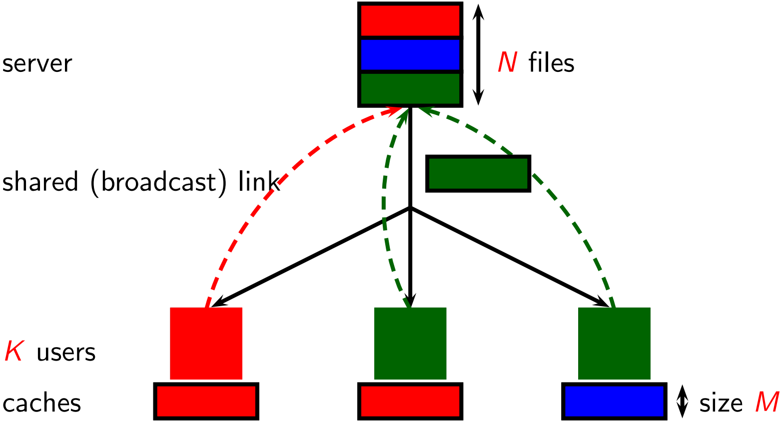
Problem Setting



Delivery: - requests are revealed to server

- server sends arbitrary function of files

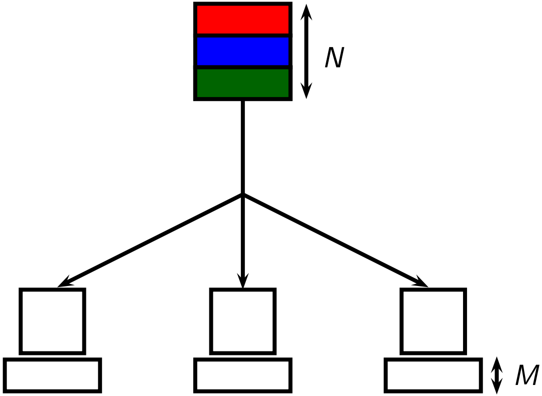
Problem Setting



Question: smallest worst-case rate $R(M)$ needed in delivery phase?

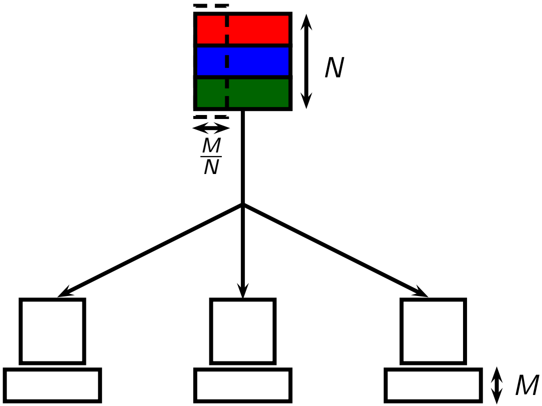
Uncoded Caching Scheme

N files, K users, cache size M



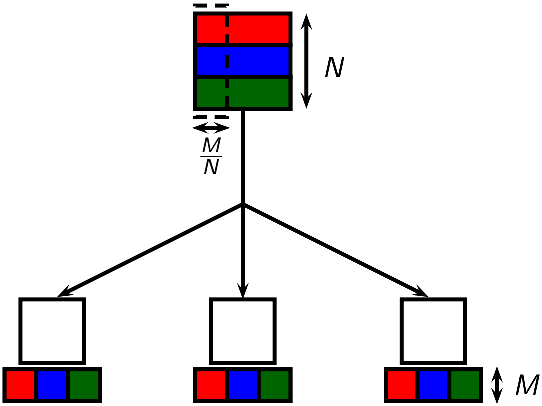
Uncoded Caching Scheme

N files, K users, cache size M



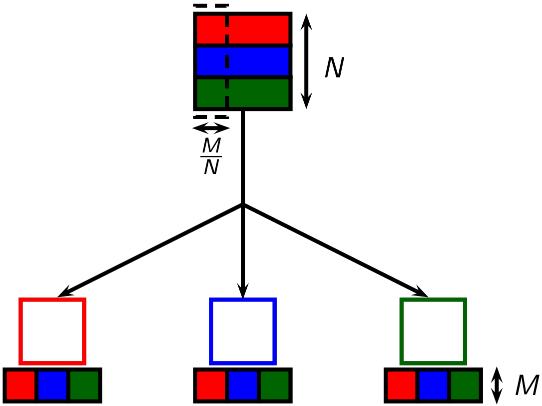
Uncoded Caching Scheme

N files, K users, cache size M



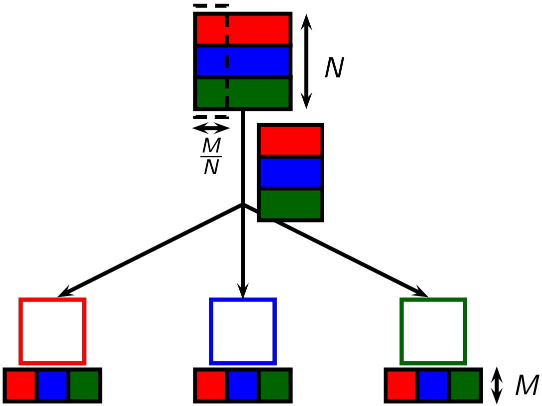
Uncoded Caching Scheme

N files, K users, cache size M



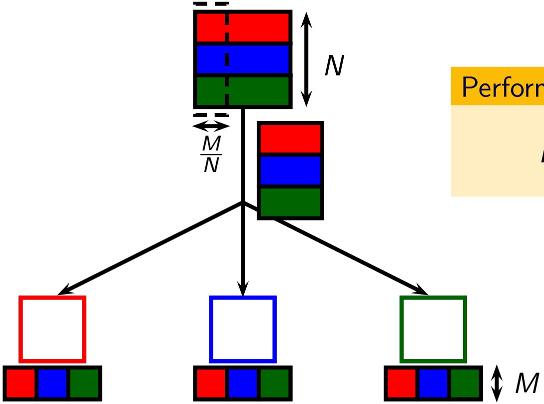
Uncoded Caching Scheme

N files, K users, cache size M



Uncoded Caching Scheme

N files, K users, cache size M

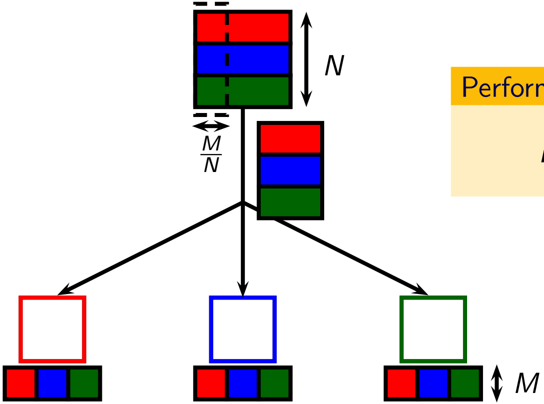


Performance of uncoded scheme:

$$R(M) = K \cdot (1 - M/N)$$

Uncoded Caching Scheme

N files, K users, cache size M



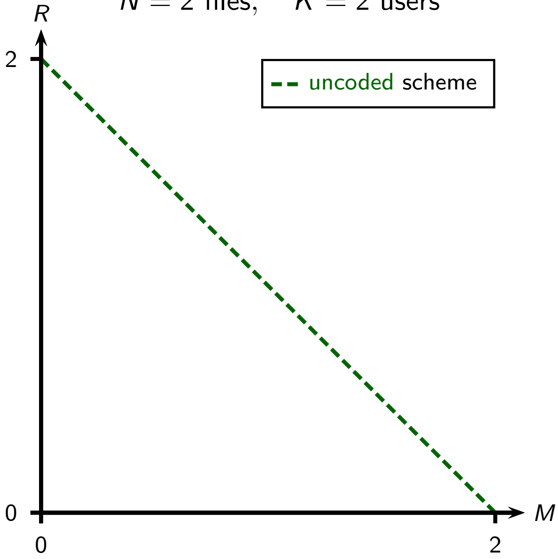
Performance of uncoded scheme:

$$R(M) = K \cdot (1 - M/N)$$

- Caches provide content locally $\Rightarrow$ local cache size matters
- Identical cache content at users

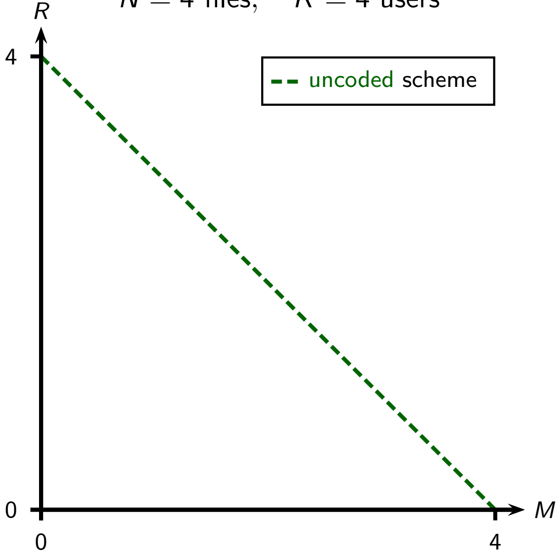
Uncoded Caching Scheme

$N = 2$ files, $\quad K = 2$ users

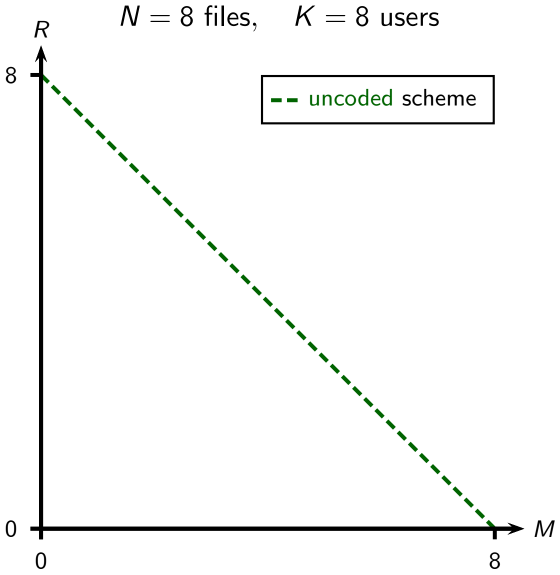


Uncoded Caching Scheme

$N = 4$ files, $\quad K = 4$ users

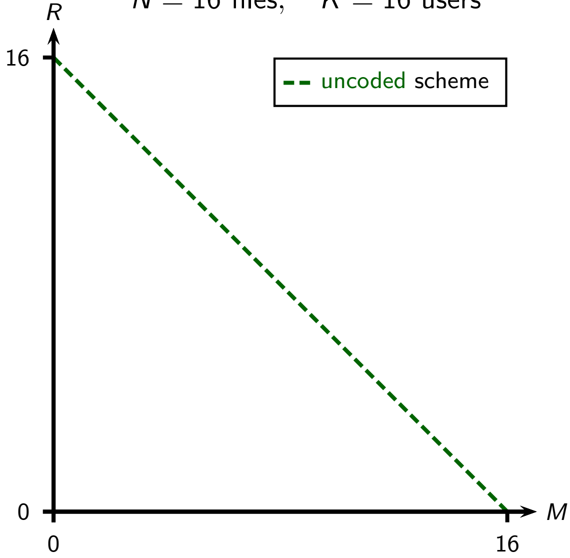


Uncoded Caching Scheme



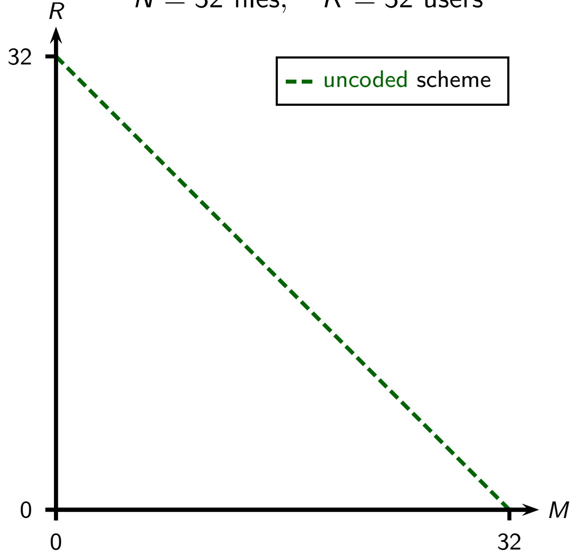
Uncoded Caching Scheme

$N = 16$ files, $K = 16$ users



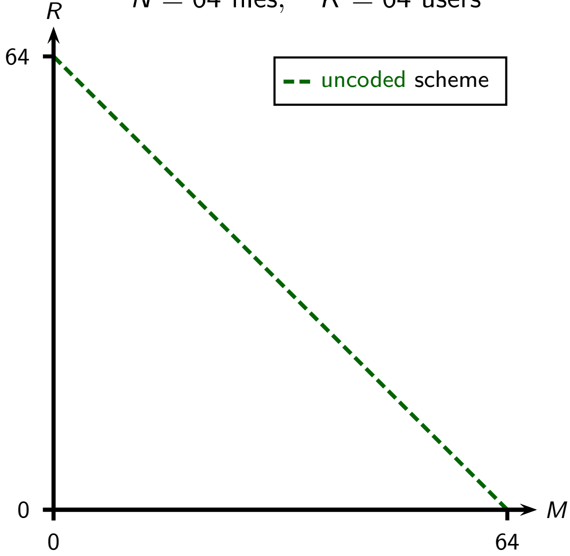
Uncoded Caching Scheme

$N = 32$ files, $K = 32$ users



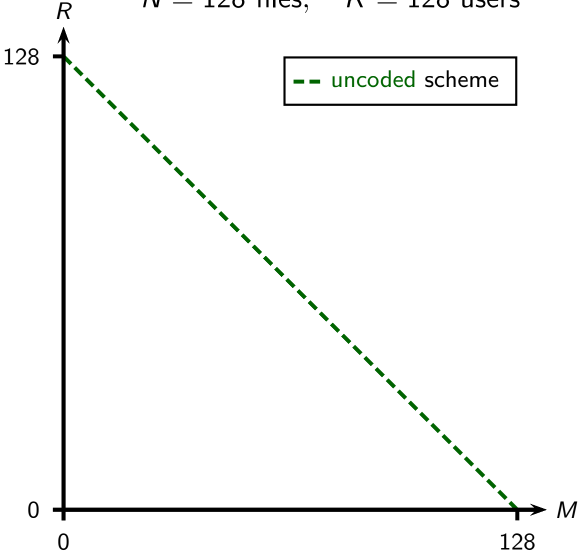
Uncoded Caching Scheme

$N = 64$ files, $\quad K = 64$ users



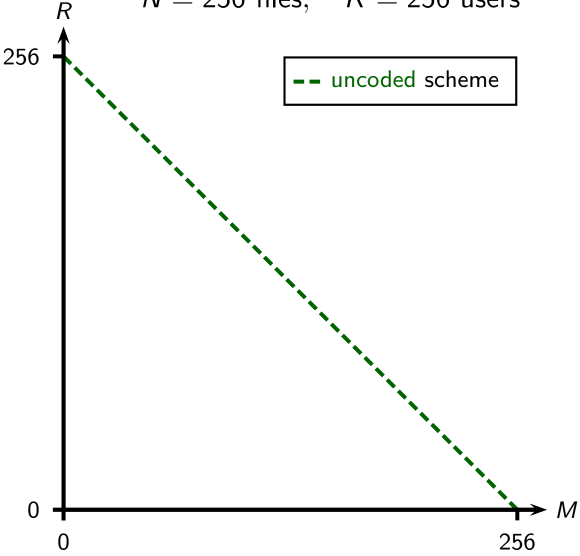
Uncoded Caching Scheme

$N = 128$ files, $K = 128$ users



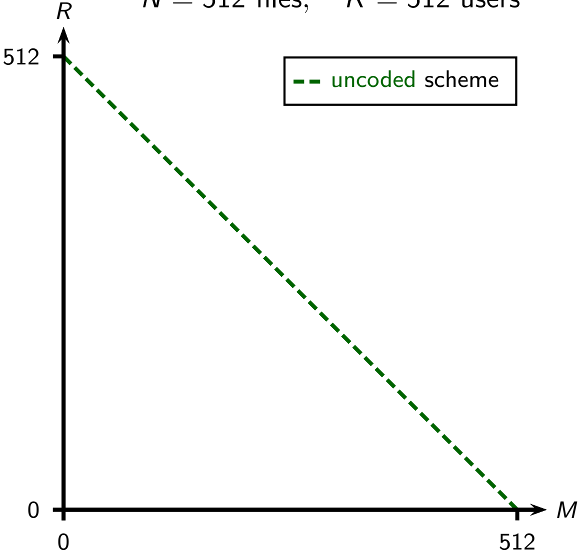
Uncoded Caching Scheme

$N = 256$ files, $K = 256$ users



Uncoded Caching Scheme

$N = 512$ files, $K = 512$ users



Proposed Coded Caching Scheme

N files, *K* users, cache size *M*

Design guidelines advocated in this talk:

- The main gain in caching is global
- Global cache size matters
- Different cache content at users
- Coded multicasting

[2] M. A. Maddah-Ali and U. Niesen, "Fundamental limits of caching," *IEEE Trans. Inf. Theory*, vol. 60, no. 5, pp. 2856–2867, May 2014.

Proposed Coded Caching Scheme

N files, *K* users, cache size *M*

Design guidelines advocated in this talk:

- The main gain in caching is global
- Global cache size matters
- Different cache content at users
- Coded multicasting

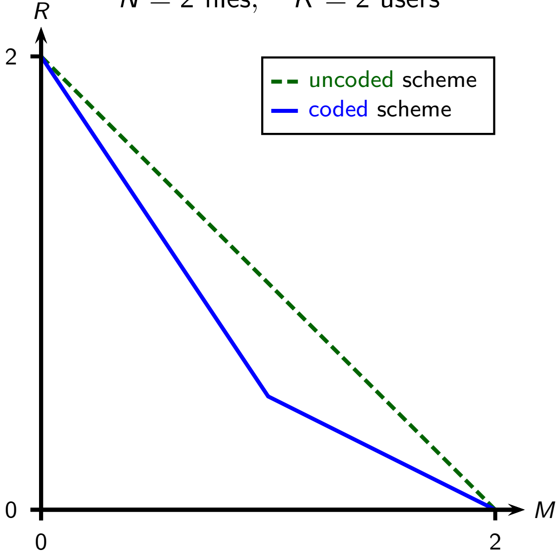
Performance of coded scheme:[2]

$$R(M) = K \cdot (1 - M/N) \cdot \frac{1}{1 + KM/N}$$

[2] M. A. Maddah-Ali and U. Niesen, "Fundamental limits of caching," *IEEE Trans. Inf. Theory*, vol. 60, no. 5, pp. 2856–2867, May 2014.

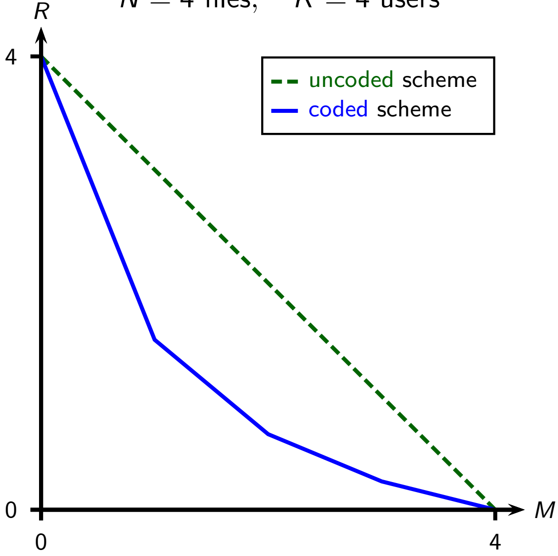
Proposed Coded Caching Scheme

$N = 2$ files, $\quad K = 2$ users



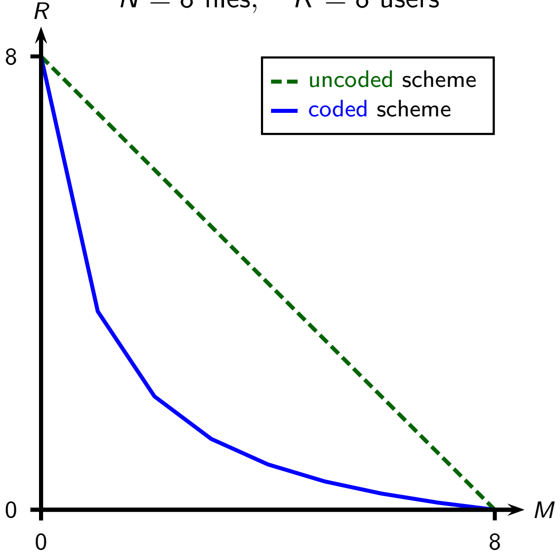
Proposed Coded Caching Scheme

$N = 4$ files, $K = 4$ users



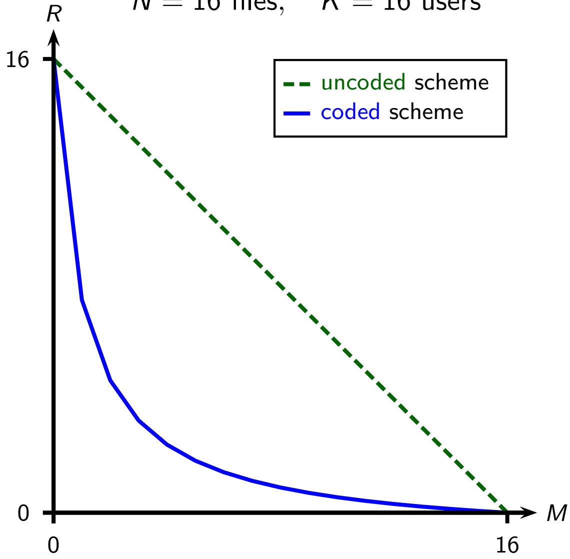
Proposed Coded Caching Scheme

$N = 8$ files, $K = 8$ users



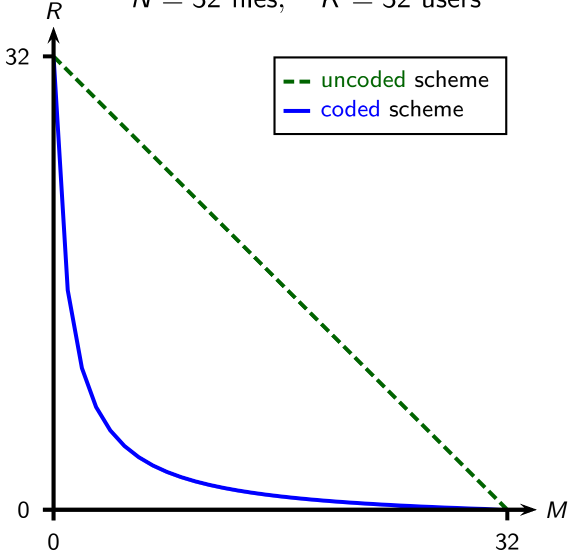
Proposed Coded Caching Scheme

$N = 16$ files, $K = 16$ users



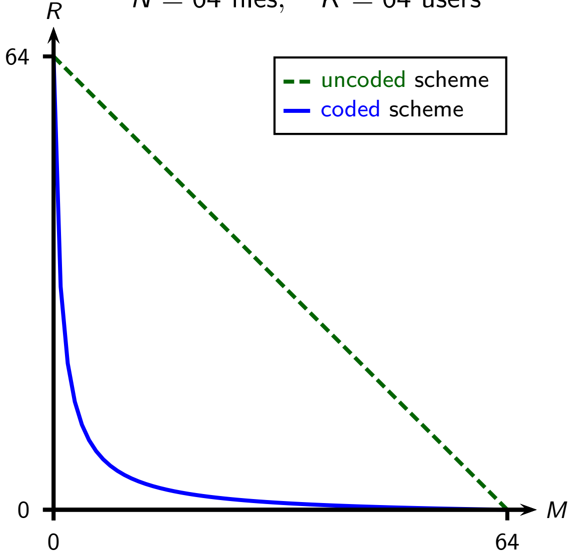
Proposed Coded Caching Scheme

$N = 32$ files, $K = 32$ users



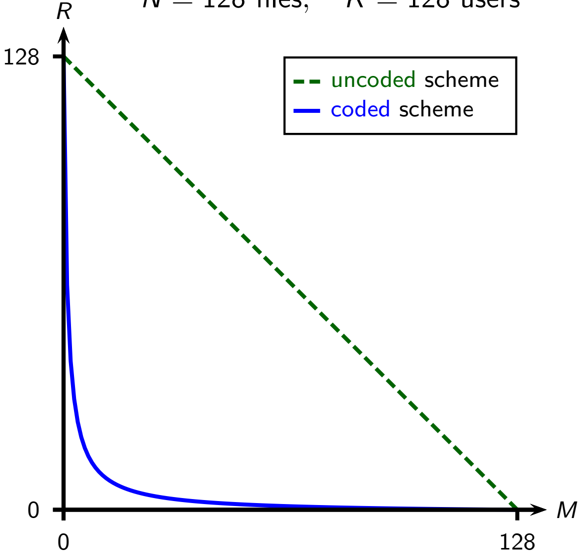
Proposed Coded Caching Scheme

$N = 64$ files, $K = 64$ users



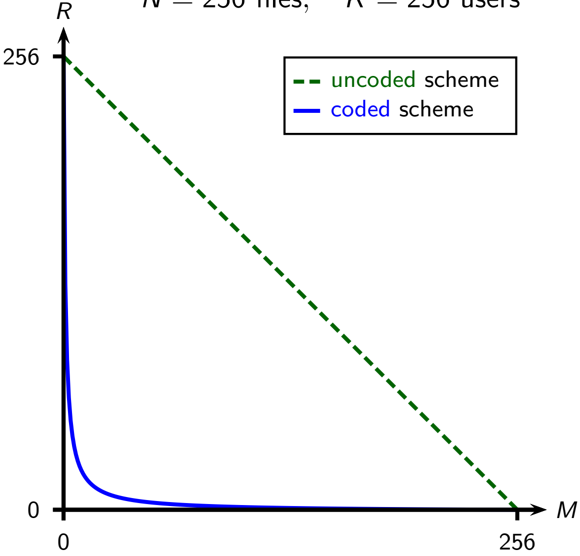
Proposed Coded Caching Scheme

$N = 128$ files, $K = 128$ users



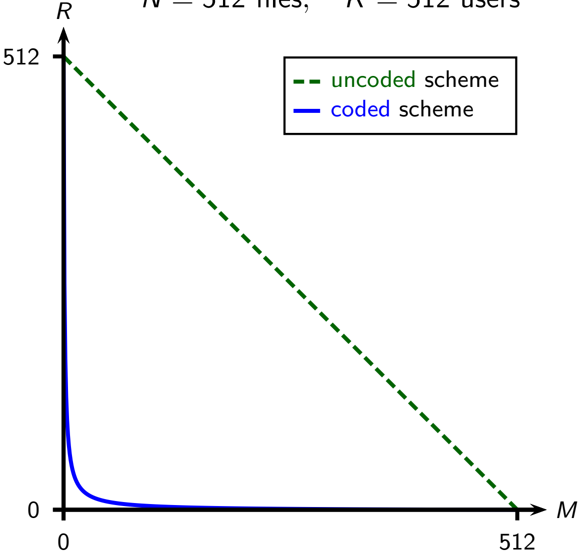
Proposed Coded Caching Scheme

$N = 256$ files, $K = 256$ users



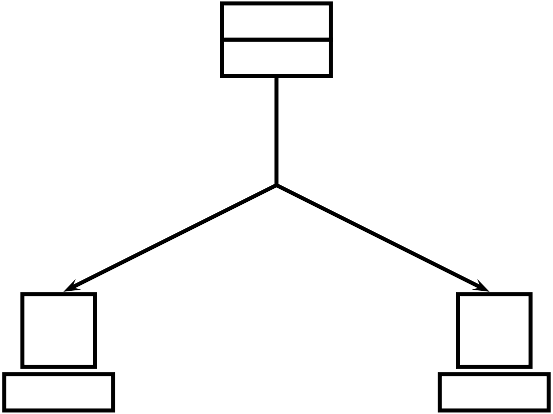
Proposed Coded Caching Scheme

$N = 512$ files, $K = 512$ users



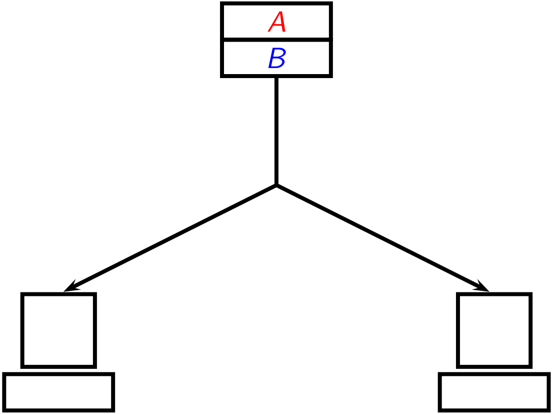
Recall: Uncoded Scheme

$N = 2$ files, $K = 2$ users, cache size $M = 1$



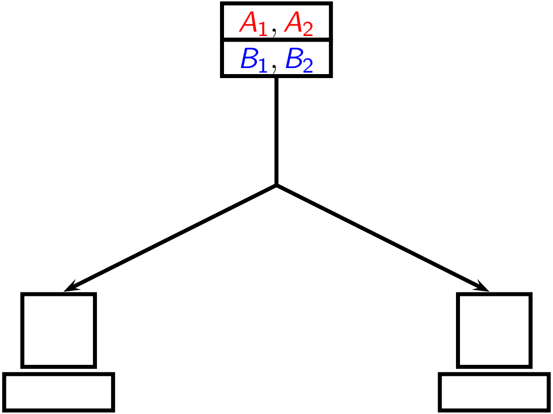
Recall: Uncoded Scheme

$N = 2$ files, $K = 2$ users, cache size $M = 1$



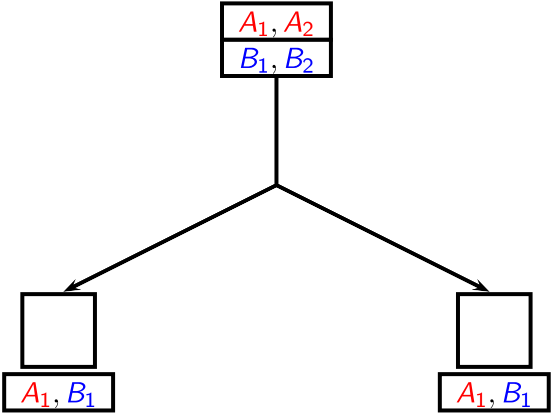
Recall: Uncoded Scheme

$N = 2$ files, $K = 2$ users, cache size $M = 1$



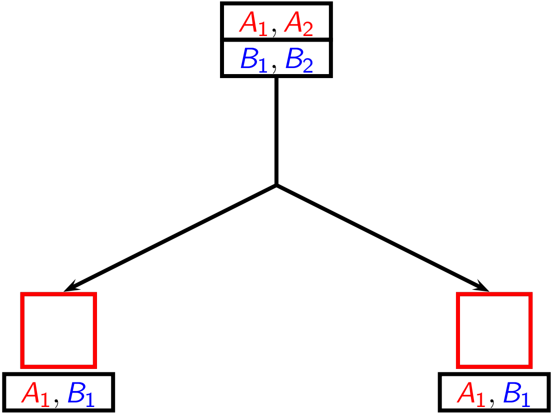
Recall: Uncoded Scheme

$N = 2$ files, $K = 2$ users, cache size $M = 1$



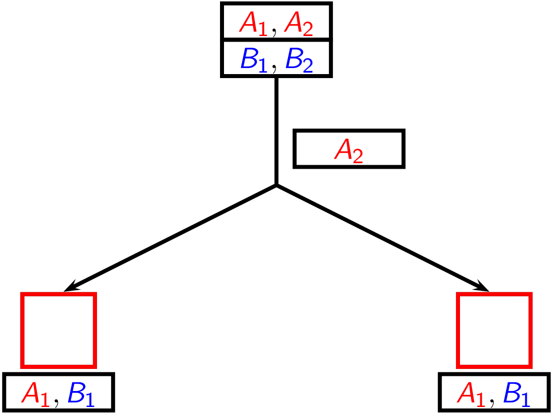
Recall: Uncoded Scheme

$N = 2$ files, $K = 2$ users, cache size $M = 1$



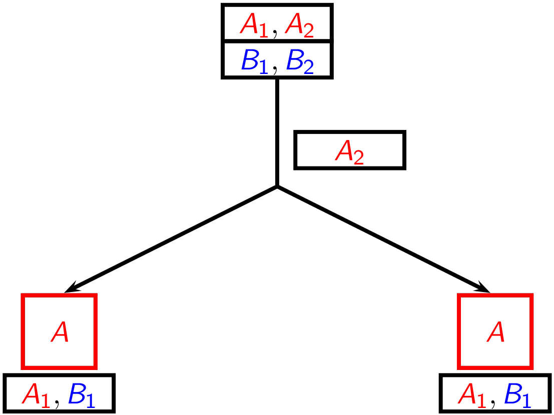
Recall: Uncoded Scheme

$N = 2$ files, $K = 2$ users, cache size $M = 1$



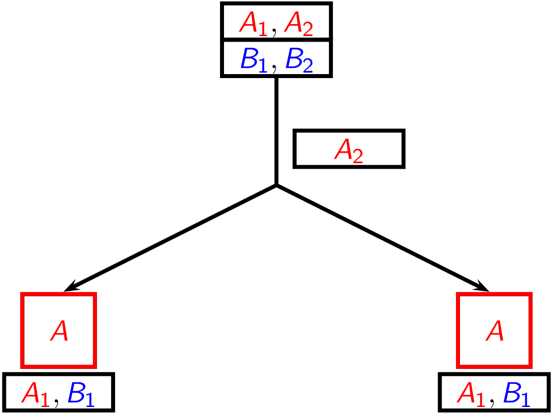
Recall: Uncoded Scheme

$N = 2$ files, $K = 2$ users, cache size $M = 1$



Recall: Uncoded Scheme

$N = 2$ files, $K = 2$ users, cache size $M = 1$

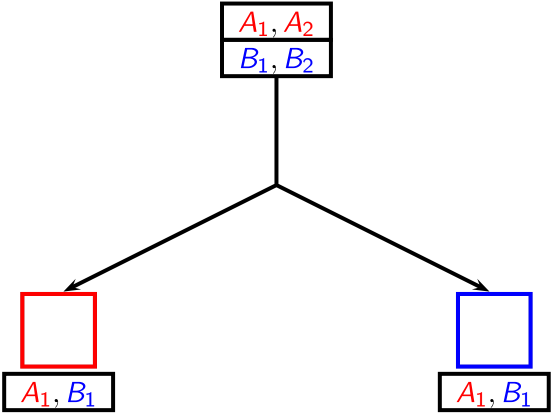


$\Rightarrow$ Identical cache content at users

$\Rightarrow$ Gain from delivering content locally

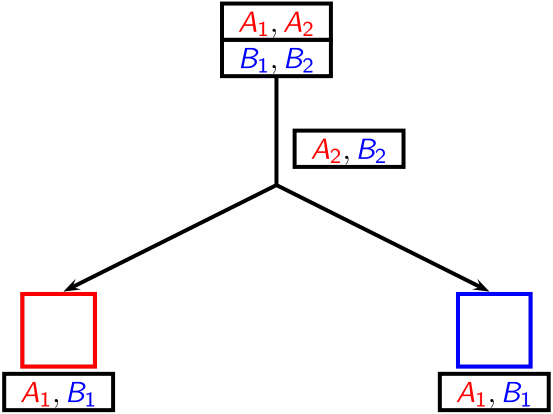
Recall: Uncoded Scheme

$N = 2$ files, $K = 2$ users, cache size $M = 1$



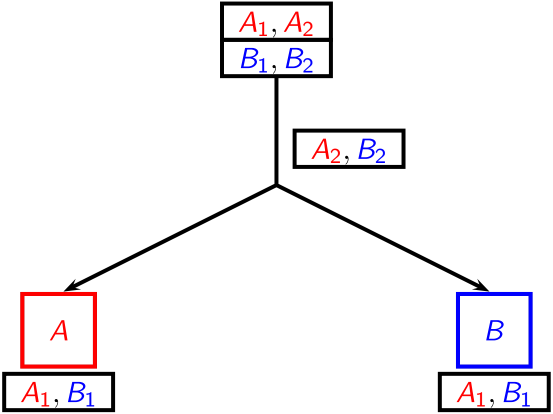
Recall: Uncoded Scheme

$N = 2$ files, $K = 2$ users, cache size $M = 1$



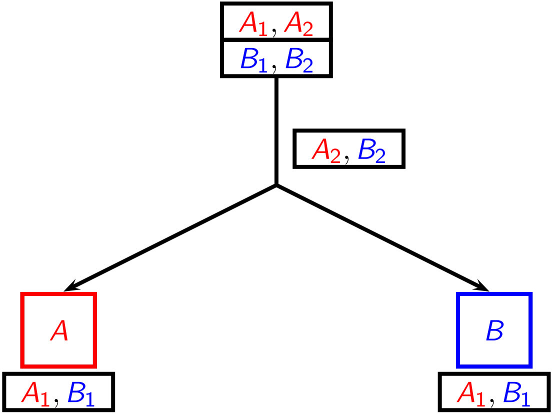
Recall: Uncoded Scheme

$N = 2$ files, $K = 2$ users, cache size $M = 1$



Recall: Uncoded Scheme

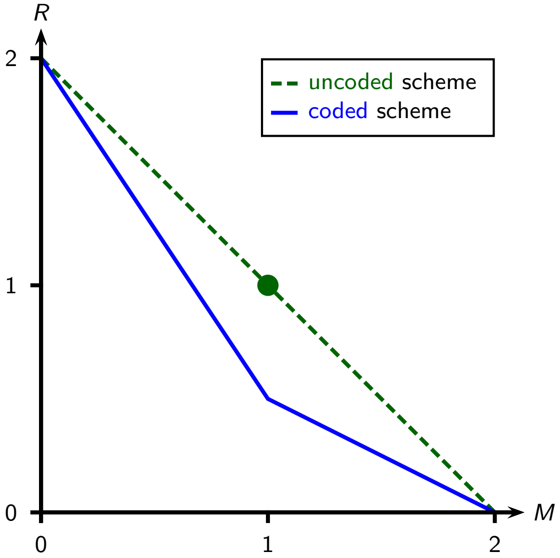
$N = 2$ files, $K = 2$ users, cache size $M = 1$



$\Rightarrow$ Multicast only possible for users with same demand

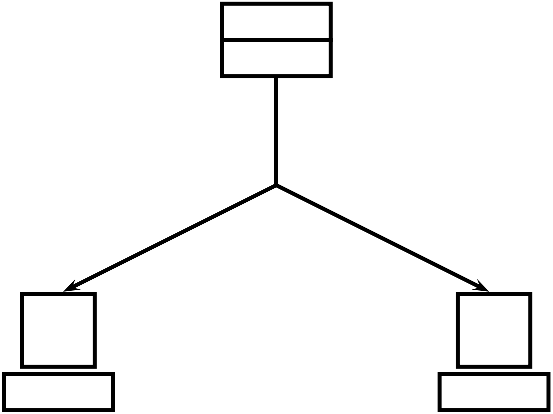
Recall: Uncoded Scheme

$N = 2$ files, $K = 2$ users, cache size $M = 1$



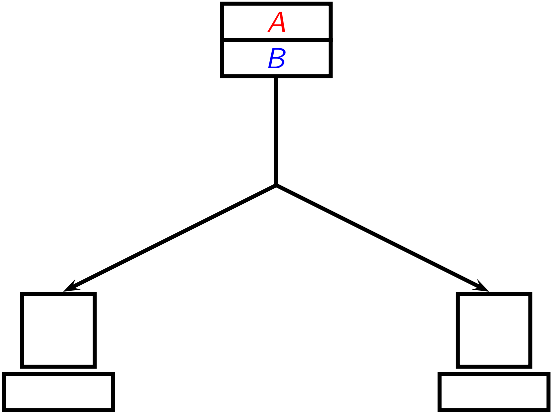
Proposed Coded Scheme

$N = 2$ files, $K = 2$ users, cache size $M = 1$



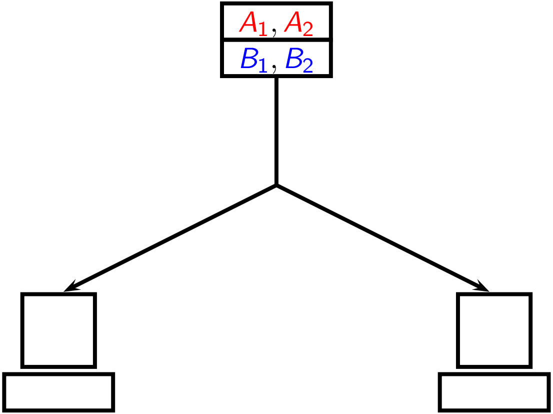
Proposed Coded Scheme

$N = 2$ files, $K = 2$ users, cache size $M = 1$



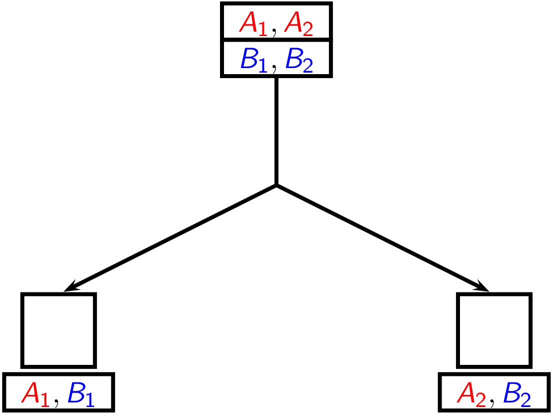
Proposed Coded Scheme

$N = 2$ files, $K = 2$ users, cache size $M = 1$



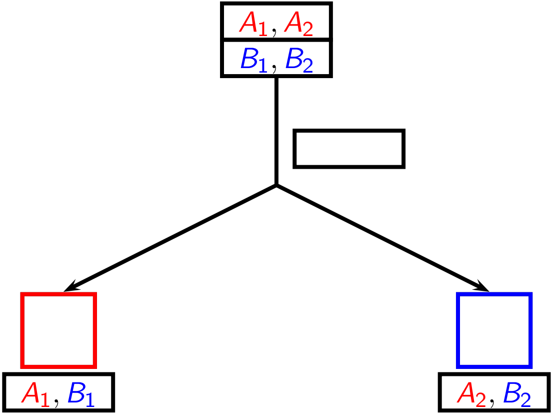
Proposed Coded Scheme

$N = 2$ files, $K = 2$ users, cache size $M = 1$



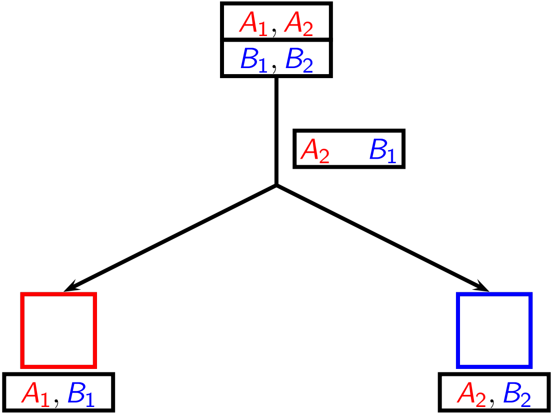
Proposed Coded Scheme

$N = 2$ files, $K = 2$ users, cache size $M = 1$



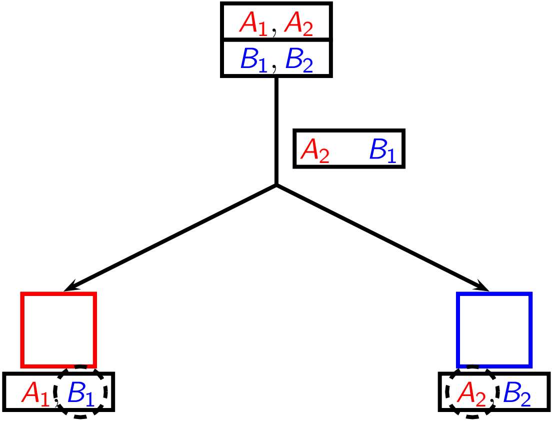
Proposed Coded Scheme

$N = 2$ files, $K = 2$ users, cache size $M = 1$



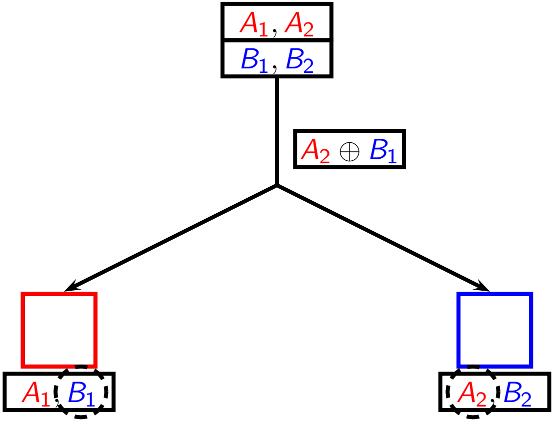
Proposed Coded Scheme

$N = 2$ files, $K = 2$ users, cache size $M = 1$



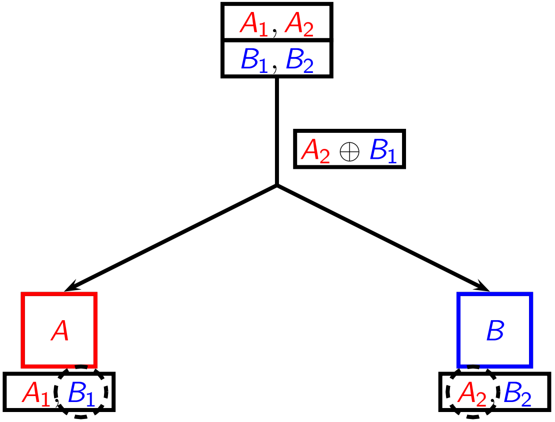
Proposed Coded Scheme

$N = 2$ files, $K = 2$ users, cache size $M = 1$



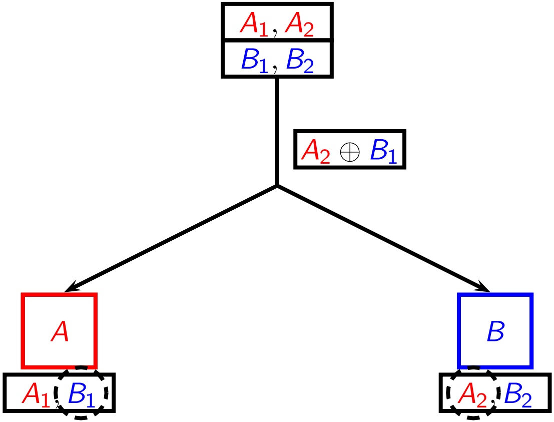
Proposed Coded Scheme

$N = 2$ files, $K = 2$ users, cache size $M = 1$



Proposed Coded Scheme

$N = 2$ files, $K = 2$ users, cache size $M = 1$

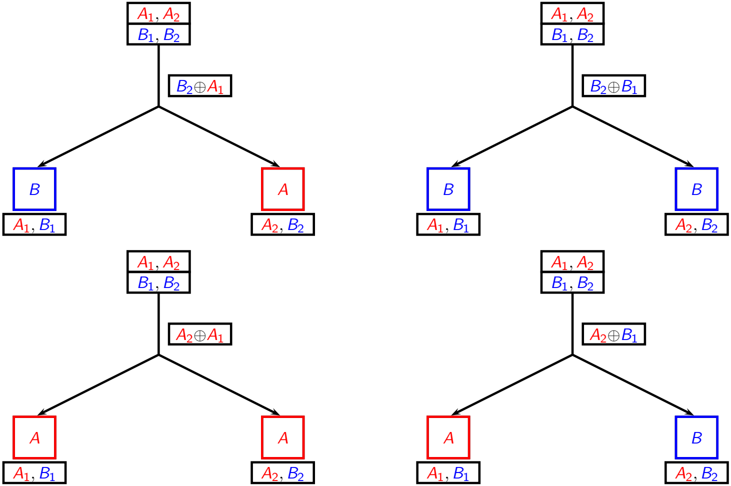


$\Rightarrow$ Different cache content at users

$\Rightarrow$ Coded multicast to 2 users with different demands

Proposed Coded Scheme

$N = 2$ files, $K = 2$ users, cache size $M = 1$

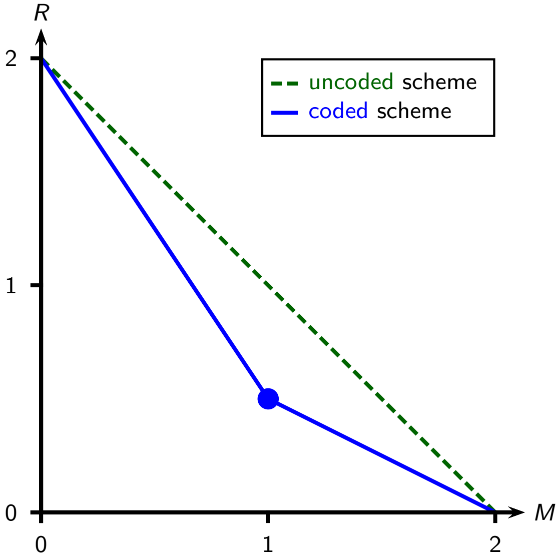


$\Rightarrow$ Works for all possible user requests

$\Rightarrow$ Simultaneous coded multicasting gain

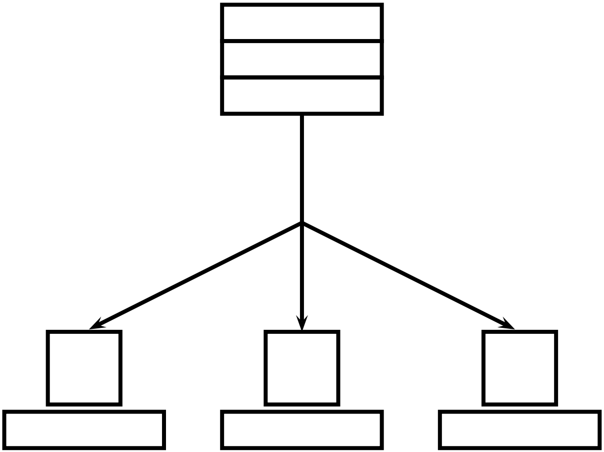
Proposed Coded Scheme

$N = 2$ files, $K = 2$ users, cache size $M = 1$



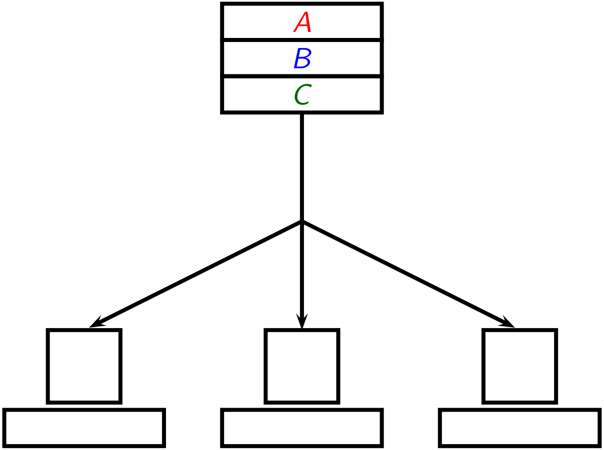
Proposed Coded Scheme

$N = 3$ files, $K = 3$ users, cache size $M = 1$



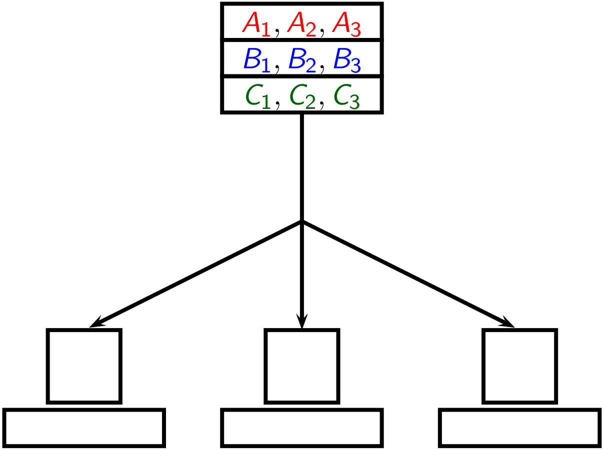
Proposed Coded Scheme

$N = 3$ files, $K = 3$ users, cache size $M = 1$



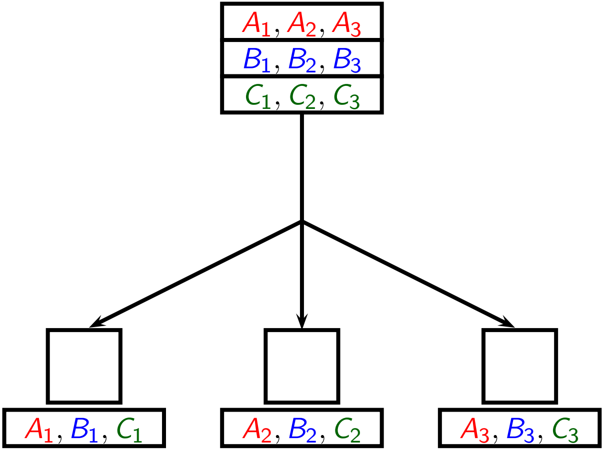
Proposed Coded Scheme

$N = 3$ files, $K = 3$ users, cache size $M = 1$



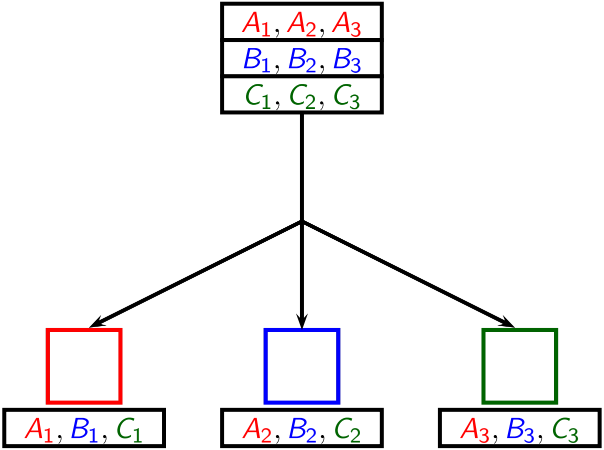
Proposed Coded Scheme

$N = 3$ files, $K = 3$ users, cache size $M = 1$



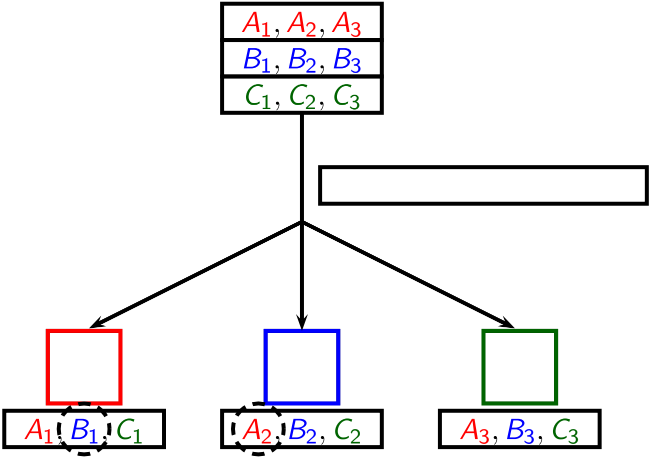
Proposed Coded Scheme

$N = 3$ files, $K = 3$ users, cache size $M = 1$



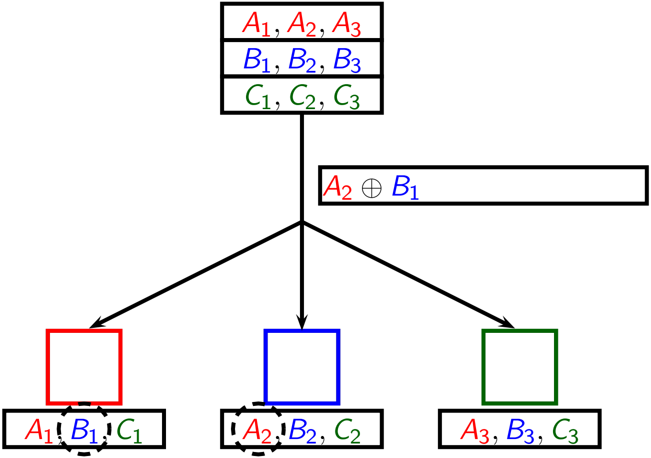
Proposed Coded Scheme

$N = 3$ files, $K = 3$ users, cache size $M = 1$



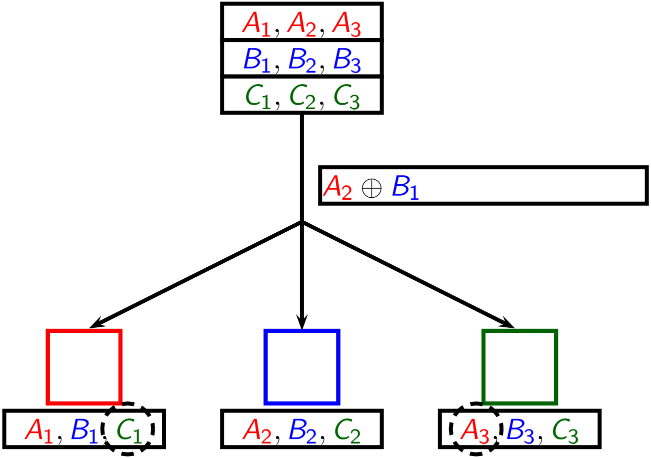
Proposed Coded Scheme

$N = 3$ files, $K = 3$ users, cache size $M = 1$



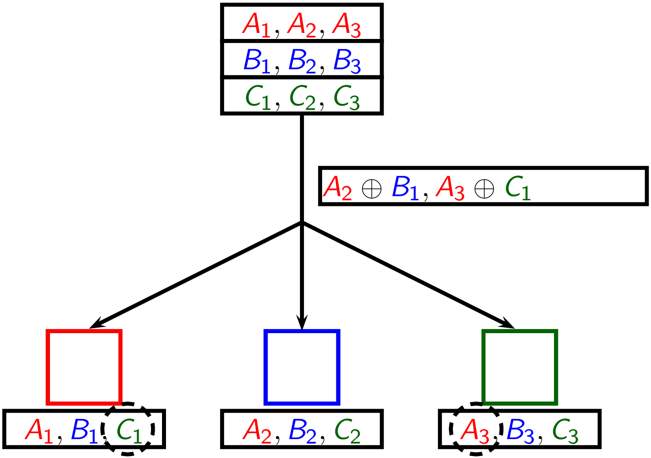
Proposed Coded Scheme

$N = 3$ files, $K = 3$ users, cache size $M = 1$



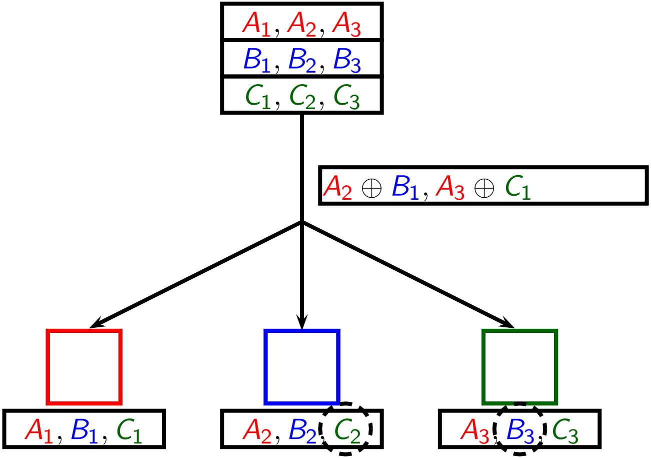
Proposed Coded Scheme

$N = 3$ files, $K = 3$ users, cache size $M = 1$



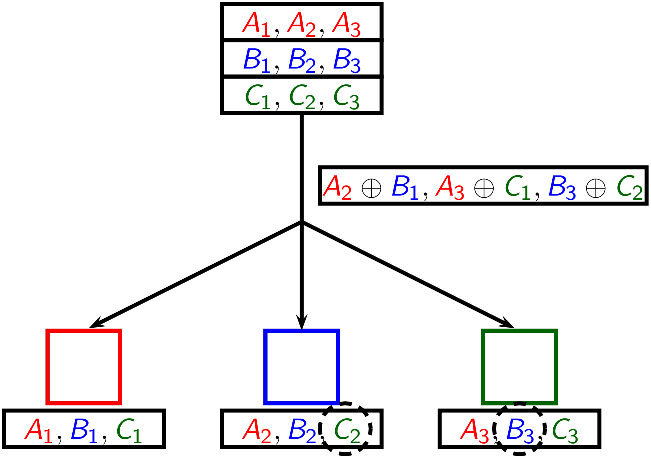
Proposed Coded Scheme

$N = 3$ files, $K = 3$ users, cache size $M = 1$



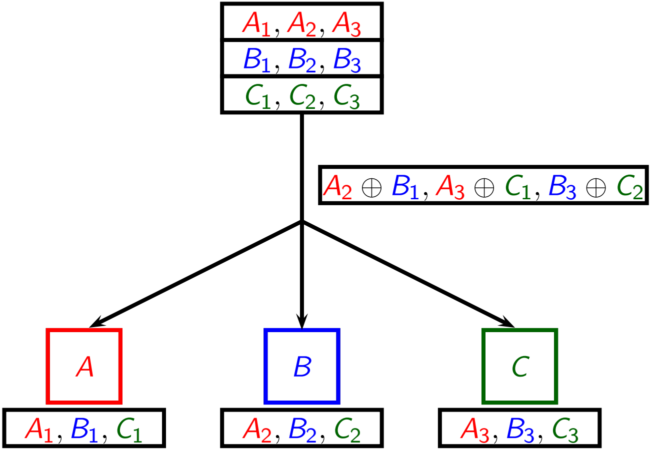
Proposed Coded Scheme

$N = 3$ files, $K = 3$ users, cache size $M = 1$



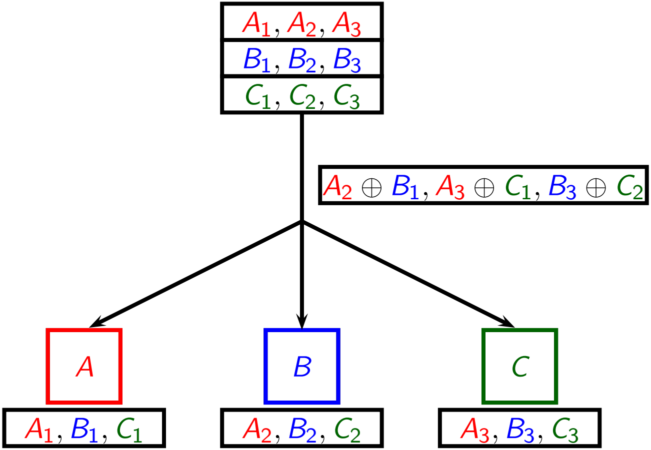
Proposed Coded Scheme

$N = 3$ files, $K = 3$ users, cache size $M = 1$



Proposed Coded Scheme

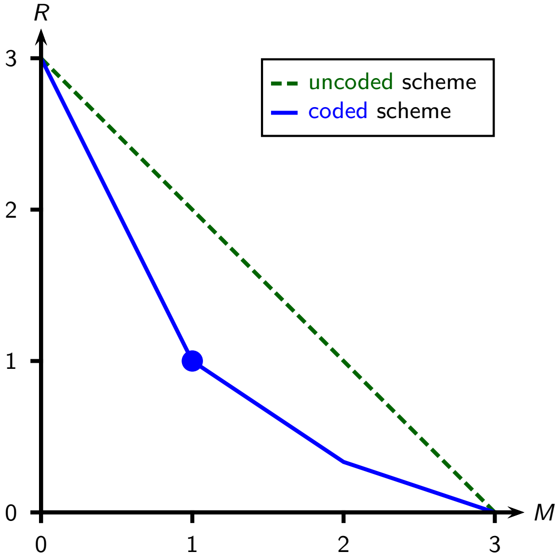
$N = 3$ files, $K = 3$ users, cache size $M = 1$



$\Rightarrow$ Coded multicast to 2 users with different demands

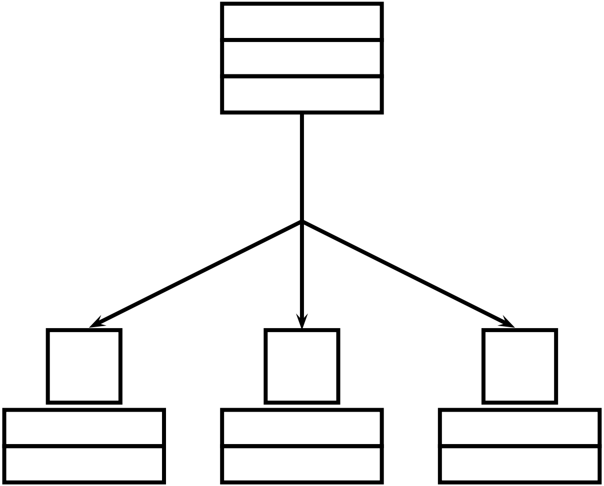
Proposed Coded Scheme

$N = 3$ files, $K = 3$ users, cache size $M = 1$



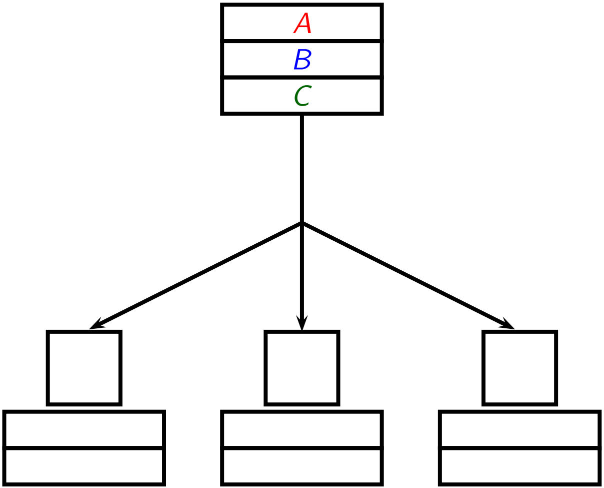
Proposed Coded Scheme

$N = 3$ files, $K = 3$ users, cache size $M = 2$



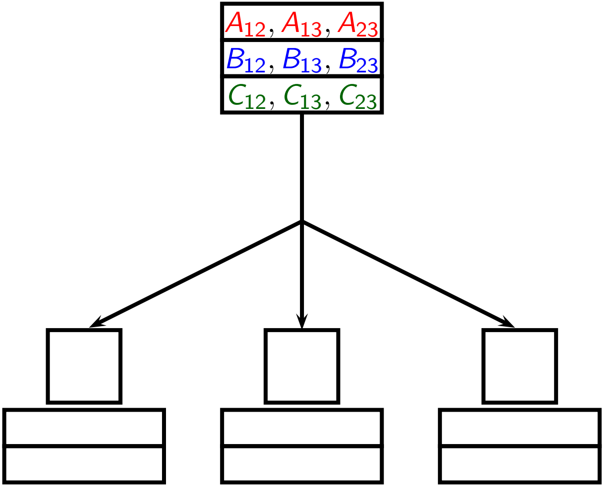
Proposed Coded Scheme

$N = 3$ files, $K = 3$ users, cache size $M = 2$



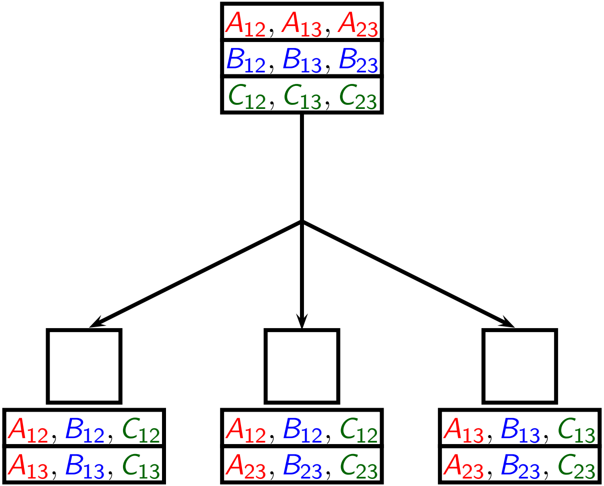
Proposed Coded Scheme

$N = 3$ files, $K = 3$ users, cache size $M = 2$



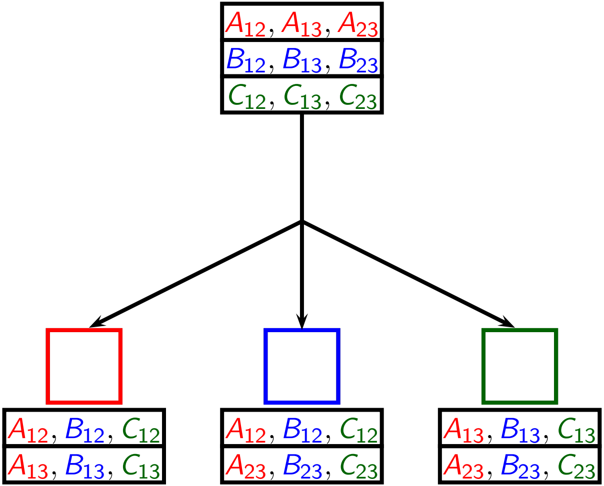
Proposed Coded Scheme

$N = 3$ files, $K = 3$ users, cache size $M = 2$



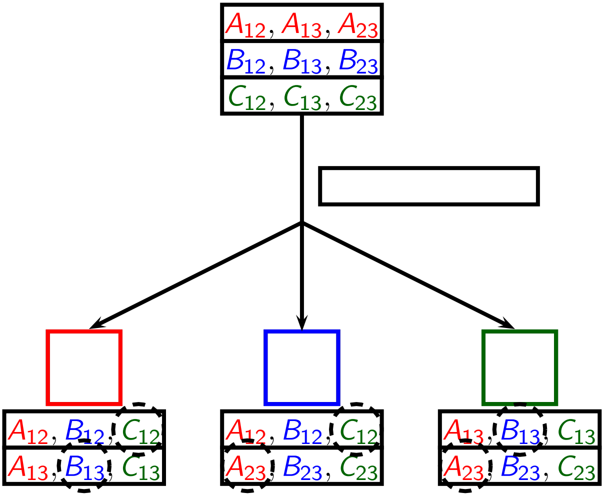
Proposed Coded Scheme

$N = 3$ files, $K = 3$ users, cache size $M = 2$



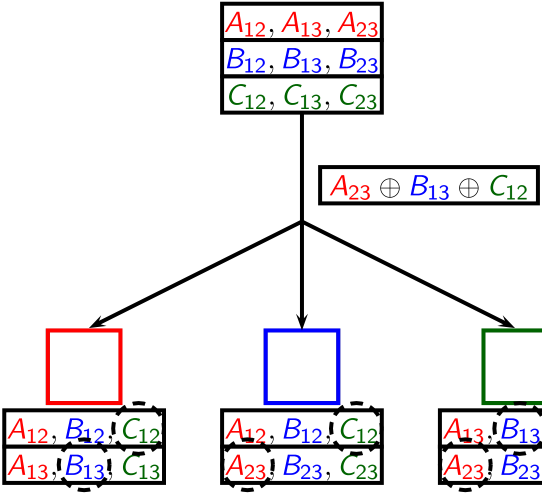
Proposed Coded Scheme

$N = 3$ files, $K = 3$ users, cache size $M = 2$



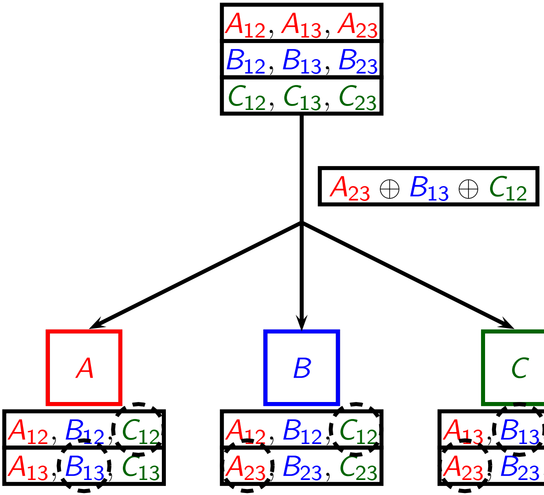
Proposed Coded Scheme

$N = 3$ files, $K = 3$ users, cache size $M = 2$



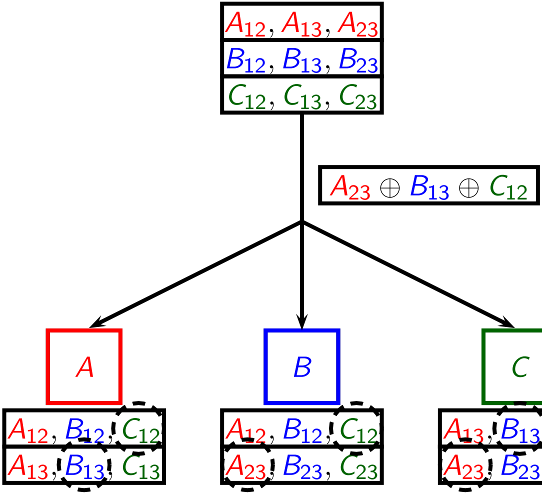
Proposed Coded Scheme

$N = 3$ files, $K = 3$ users, cache size $M = 2$



Proposed Coded Scheme

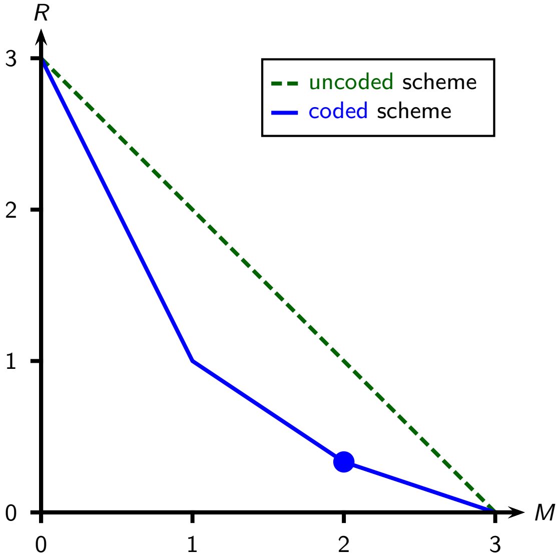
$N = 3$ files, $K = 3$ users, cache size $M = 2$



$\Rightarrow$ Coded multicast to 3 users with different demands

Proposed Coded Scheme

$N = 3$ files, $K = 3$ users, cache size $M = 2$



Proposed Coded Scheme

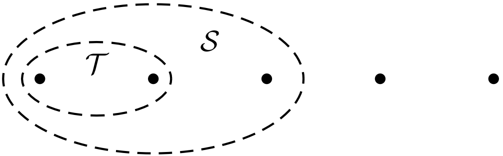
$N = K$ files and users, cache size M

- Goal: coded multicast to $M + 1$ users with different demands

Proposed Coded Scheme

$N = K$ files and users, cache size M

- Goal: coded multicast to $M + 1$ users with different demands
- Need to place content such that in delivery phase:
  1. for every possible user demands...
  2. and for every possible subset $\mathcal{S}$ of $M + 1$ users...
  3. and for every possible subset $\mathcal{T} \subset \mathcal{S}$ of M users...
  4. users in $\mathcal{T}$ share content that is required at the user in $\mathcal{S} \setminus \mathcal{T}$



Proposed Coded Scheme

$N = K$ files and users, cache size M

- Goal: coded multicast to $M + 1$ users with different demands
- Need to place content such that in delivery phase:
  1. for every possible user demands. . .
  2. and for every possible subset $\mathcal{S}$ of $M + 1$ users. . .
  3. and for every possible subset $\mathcal{T} \subset \mathcal{S}$ of M users. . .
  4. users in $\mathcal{T}$ share content that is required at the user in $\mathcal{S} \setminus \mathcal{T}$

Example: $N = K = 3$, $M = 2$

Every two users have a piece of content the remaining user needs

Proposed Coded Scheme
$N = K$ files and users, cache size M

Placement phase:

Proposed Coded Scheme

$N = K$ files and users, cache size M

Placement phase:

- N files: $W_1, \ldots, W_N$

Proposed Coded Scheme

$N = K$ files and users, cache size M

Placement phase:

- N files: $W_1, \ldots, W_N$
- Split each file into $\binom{K}{M}$ parts

  $\Rightarrow\ W_n = \big(W_{n,\mathcal{T}} : \mathcal{T} \subset [K], |\mathcal{T}| = M\big)$

Proposed Coded Scheme

$N = K$ files and users, cache size M

Placement phase:

- N files: $W_1, \ldots, W_N$
- Split each file into $\binom{K}{M}$ parts
  $\Rightarrow$ $W_n = \big(W_{n,\mathcal{T}} : \mathcal{T} \subset [K], |\mathcal{T}| = M\big)$
- Cache k: $\big(W_{n,\mathcal{T}} : n \in [N], \mathcal{T} \subset [K], |\mathcal{T}| = M, k \in \mathcal{T}\big)$

Proposed Coded Scheme

$N = K$ files and users, cache size M

Placement phase:

- N files: $W_1, \ldots, W_N$
- Split each file into $\binom{K}{M}$ parts
  $\Rightarrow$ $W_n = \big(W_{n,\mathcal{T}} : \mathcal{T} \subset [K], |\mathcal{T}| = M\big)$
- Cache k: $\big(W_{n,\mathcal{T}} : n \in [N], \mathcal{T} \subset [K], |\mathcal{T}| = M, k \in \mathcal{T}\big)$

Example: $N = K = 3$, $M = 2$

Consider files A, B, C $\Rightarrow$ cache 2: $\big(A_{12}, A_{23}, B_{12}, B_{23}, C_{12}, C_{23}\big)$

Proposed Coded Scheme

$N = K$ files and users, cache size M

Delivery phase:

Proposed Coded Scheme

$N = K$ files and users, cache size M

Delivery phase: Assume users k requests W_{d_k}

Proposed Coded Scheme

$N = K$ files and users, cache size M

Delivery phase: Assume users k requests W_{d_k}

- Send $\oplus_{k \in \mathcal{S}} W_{d_k, \mathcal{S} \setminus \{k\}}$ for all $\mathcal{S} \subset [K]$ such that $|\mathcal{S}| = M + 1$

Proposed Coded Scheme

$N = K$ files and users, cache size M

Delivery phase: Assume users k requests W_{d_k}

- Send $\oplus_{k \in \mathcal{S}} W_{d_k, \mathcal{S} \setminus \{k\}}$ for all $\mathcal{S} \subset [K]$ such that $|\mathcal{S}| = M + 1$
- Coded multicast to $M + 1$ users with different demands

Proposed Coded Scheme

$N = K$ files and users, cache size M

Delivery phase: Assume users k requests W_{d_k}

- Send $\oplus_{k \in \mathcal{S}} W_{d_k, \mathcal{S} \setminus \{k\}}$ for all $\mathcal{S} \subset [K]$ such that $|\mathcal{S}| = M + 1$
- Coded multicast to $M + 1$ users with different demands

Example: $N = K = 3$, $M = 1$

Consider files A, B, C and user requests $d_1 = A, d_2 = B, d_3 = C$

$\Rightarrow$ Server sends $A_2 \oplus B_1,\ A_3 \oplus C_1,\ B_3 \oplus C_2$

Comparison of the Two Schemes

N files, K users, cache size M

- Uncoded scheme: $R(M) = K \cdot (1 - M/N)$
- Coded scheme: $R(M) = K \cdot (1 - M/N) \cdot \frac{1}{1+KM/N}$

Comparison of the Two Schemes

N files, K users, cache size M

- Uncoded scheme: $R(M) = K \cdot (1 - M/N)$
- Coded scheme: $R(M) = K \cdot (1 - M/N) \cdot \frac{1}{1+KM/N}$

- Rate without caching K

Comparison of the Two Schemes

N files, K users, cache size M

- Uncoded scheme: $R(M) = K \cdot (1 - M/N)$
- Coded scheme: $R(M) = K \cdot (1 - M/N) \cdot \frac{1}{1+KM/N}$

- Rate without caching K
- Local caching gain $1 - M/N$
  - Significant when local cache size M is of order N

Comparison of the Two Schemes

N files, K users, cache size M

- Uncoded scheme: $R(M) = K \cdot (1 - M/N)$
- Coded scheme: $R(M) = K \cdot (1 - M/N) \cdot \frac{1}{1+KM/N}$

- Rate without caching K
- Local caching gain $1 - M/N$
  - Significant when local cache size M is of order N
- Global caching gain $\frac{1}{1+KM/N}$
  - Significant when global cache size KM is of order N

Comparison of the Two Schemes

N files, K users, cache size M

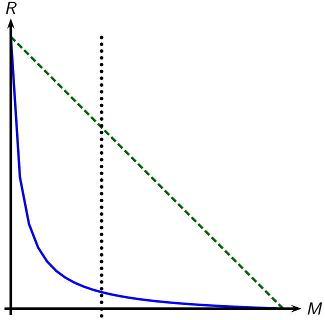
- Uncoded scheme: $R(M) = K \cdot (1 - M/N)$
- Coded scheme: $R(M) = K \cdot (1 - M/N) \cdot \frac{1}{1+KM/N}$

- Rate without caching K
- Local caching gain $1 - M/N$
  - Significant when local cache size M is of order N
- Global caching gain $\frac{1}{1+KM/N}$
  - Significant when global cache size KM is of order N

$\Rightarrow$ Global gain can be $\Theta(K)$ smaller than local gain

Example

$N = 30$ files, $K = 30$ users, cache size $M = 10$

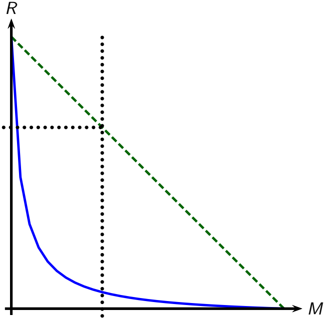


Example

$N = 30$ files, $K = 30$ users, cache size $M = 10$

- Uncoded scheme:

$$R(M) = K \cdot (1 - M/N)$$
$$\approx 30 \cdot 0.67 \approx 20$$



Example

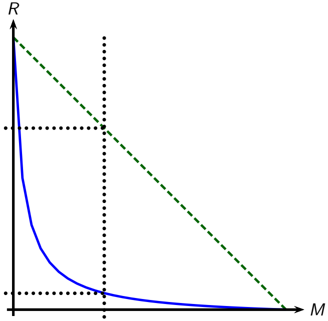
$N = 30$ files, $K = 30$ users, cache size $M = 10$

- Uncoded scheme:

$$R(M) = K \cdot (1 - M/N)$$
$$\approx 30 \cdot 0.67 \approx 20$$

- Coded scheme:

$$R(M) = K \cdot (1 - M/N) \cdot \frac{1}{1 + KM/N}$$
$$\approx 30 \cdot 0.67 \cdot 0.09 \approx 1.8$$



Example

$N = 30$ files, $K = 30$ users, cache size $M = 10$

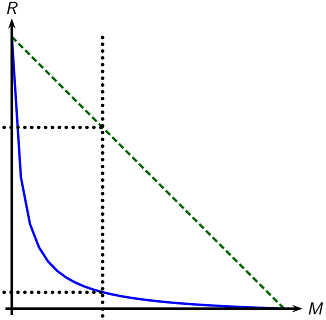
- Uncoded scheme:

$$R(M) = K \cdot (1 - M/N)$$
$$\approx 30 \cdot 0.67 \approx 20$$

- Coded scheme:

$$R(M) = K \cdot (1 - M/N) \cdot \frac{1}{1 + KM/N}$$
$$\approx 30 \cdot 0.67 \cdot 0.09 \approx 1.8$$

$\Rightarrow$ Factor 11 reduction in rate!



Example

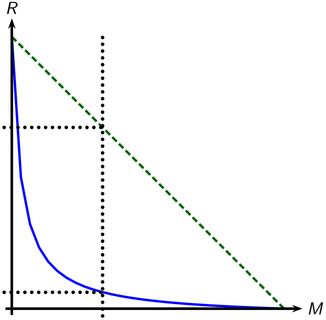
$N = 30$ files, $K = 30$ users, cache size $M = 10$

- Uncoded scheme:

$$R(M) = K \cdot (1 - M/N)$$
$$\approx 30 \cdot 0.67 \approx 20$$

- Coded scheme:

$$R(M) = K \cdot (1 - M/N) \cdot \frac{1}{1 + KM/N}$$
$$\approx 30 \cdot 0.67 \cdot 0.09 \approx 1.8$$



⇒ Factor 11 reduction in rate!

⇒ Local gain is 0.67

Example

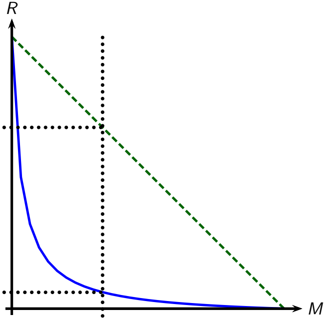
$N = 30$ files, $K = 30$ users, cache size $M = 10$

- Uncoded scheme:

$$R(M) = K \cdot (1 - M/N) \approx 30 \cdot 0.67 \approx 20$$

- Coded scheme:

$$R(M) = K \cdot (1 - M/N) \cdot \frac{1}{1 + KM/N} \approx 30 \cdot 0.67 \cdot 0.09 \approx 1.8$$



$\Rightarrow$ Factor 11 reduction in rate!

$\Rightarrow$ Local gain is 0.67

$\Rightarrow$ Global gain is 0.09
(coded multicast to $M + 1 = 11$ users with different demands)

Can We Do Better?

Theorem

The coded scheme is optimal to within a constant factor in rate.[3]

[3] M. A. Maddah-Ali and U. Niesen, "Fundamental limits of caching," *IEEE Trans. Inf. Theory*, vol. 60, no. 5, pp. 2856–2867, May 2014.

Can We Do Better?

Theorem

The coded scheme is optimal to within a constant factor in rate.[3]

$\Rightarrow$ Information-theoretic bound

[3] M. A. Maddah-Ali and U. Niesen, "Fundamental limits of caching," *IEEE Trans. Inf. Theory*, vol. 60, no. 5, pp. 2856–2867, May 2014.

Can We Do Better?

Theorem

The coded scheme is optimal to within a constant factor in rate.[3]

⇒ Information-theoretic bound

⇒ Constant is independent of problem parameters N, K, M

[3] M. A. Maddah-Ali and U. Niesen, "Fundamental limits of caching," *IEEE Trans. Inf. Theory*, vol. 60, no. 5, pp. 2856–2867, May 2014.

Can We Do Better?

Theorem

The coded scheme is optimal to within a constant factor in rate.[3]

⇒ Information-theoretic bound

⇒ Constant is independent of problem parameters N, K, M

⇒ No other significant gain besides local and global

[3] M. A. Maddah-Ali and U. Niesen, "Fundamental limits of caching," *IEEE Trans. Inf. Theory*, vol. 60, no. 5, pp. 2856–2867, May 2014.

Approach Can be Adapted to Handle...

- Asynchronous user requests[4]
- Nonuniform file popularities[5]
- Users joining and leaving the network[6]
- Several users sharing a cache[7]
- Online cache updates[8]
- More complicated network topologies[9]

---

[4] Niesen and Maddah-Ali 2015.

[5] Niesen and Maddah-Ali 2017; Ji, Tulino, Llorca, and Caire 2017; Zhang, Lin, and Wang 2018.

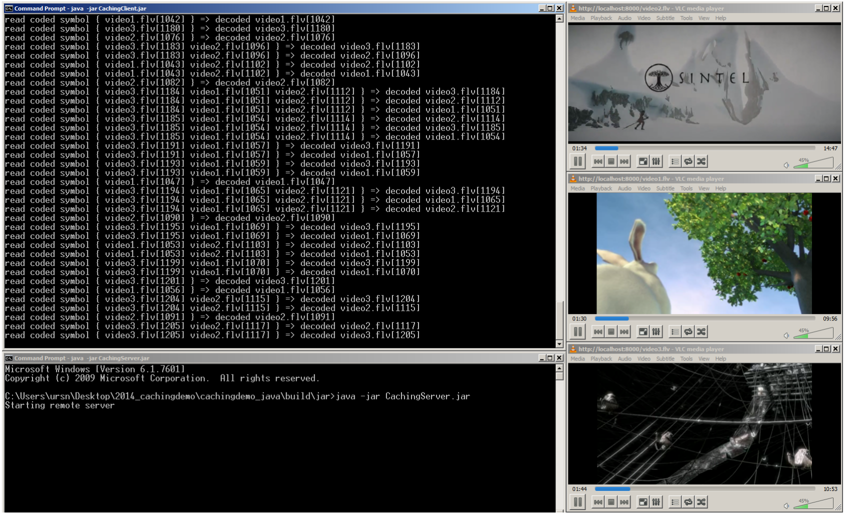
[6] Maddah-Ali and Niesen 2015.

[7] Hachem, Karamchandani, and Diggavi 2017.

[8] Pedarsani, Maddah-Ali, and Niesen 2016.

[9] Karamchandani, Niesen, Maddah-Ali, and Diggavi 2016; Ji, Caire, and Molisch 2016.

Video Streaming Demo[10]



[10] U. Niesen and M. A. Maddah-Ali, "Coded caching for delay-sensitive content," in *Proc. IEEE ICC*, Jun. 2015, pp. 5559–5564.

Open Questions

- File size scaling:

Open Questions

- File size scaling:
  - Described approach requires file size scaling like $\binom{K}{KM/N}$
  - Scales poorly with number of users K

Open Questions

- File size scaling:
  - Described approach requires file size scaling like $\binom{K}{KM/N}$
  - Scales poorly with number of users K
  - Promising recent results with interesting connection to graph theory[11], but scope for more work

[11] K. Shanmugam, A. M. Tulino, and A. G. Dimakis, "Coded caching with linear subpacketization is possible using Rusza-Szemeredi graphs," in *Proc. IEEE ISIT*, Jun. 2017, pp. 2157–8117

Open Questions

- File size scaling:
  - Described approach requires file size scaling like $\binom{K}{KM/N}$
  - Scales poorly with number of users K
  - Promising recent results with interesting connection to graph theory[11], but scope for more work
- State:

[11] K. Shanmugam, A. M. Tulino, and A. G. Dimakis, "Coded caching with linear subpacketization is possible using Rusza-Szemeredi graphs," in *Proc. IEEE ISIT*, Jun. 2017, pp. 2157–8117

Open Questions

- File size scaling:
  - Described approach requires file size scaling like $\binom{K}{KM/N}$
  - Scales poorly with number of users K
  - Promising recent results with interesting connection to graph theory[11], but scope for more work
- State:
  - Standard caching schemes only require knowledge of local state
  - In contrast coded caching requires the server to have knowledge of global state

[11] K. Shanmugam, A. M. Tulino, and A. G. Dimakis, "Coded caching with linear subpacketization is possible using Rusza-Szemeredi graphs," in *Proc. IEEE ISIT*, Jun. 2017, pp. 2157–8117

Open Questions

- File size scaling:
  - Described approach requires file size scaling like $\binom{K}{KM/N}$
  - Scales poorly with number of users K
  - Promising recent results with interesting connection to graph theory[11], but scope for more work
- State:
  - Standard caching schemes only require knowledge of local state
  - In contrast coded caching requires the server to have knowledge of global state
- Large scale implementation:

[11] K. Shanmugam, A. M. Tulino, and A. G. Dimakis, "Coded caching with linear subpacketization is possible using Rusza-Szemeredi graphs," in *Proc. IEEE ISIT*, Jun. 2017, pp. 2157–8117

Open Questions

- File size scaling:
  - Described approach requires file size scaling like $\binom{K}{KM/N}$
  - Scales poorly with number of users K
  - Promising recent results with interesting connection to graph theory[11], but scope for more work
- State:
  - Standard caching schemes only require knowledge of local state
  - In contrast coded caching requires the server to have knowledge of global state
- Large scale implementation:
  - So far only demo-sized implementation
  - Experimentation with large-scale systems are needed

[11] K. Shanmugam, A. M. Tulino, and A. G. Dimakis, "Coded caching with linear subpacketization is possible using Rusza-Szemeredi graphs," in *Proc. IEEE ISIT*, Jun. 2017, pp. 2157–8117

Conclusions

A New Approach to Caching

Conclusions
A New Approach to Caching

- Main gain in caching is global
  - $\Rightarrow$ Coded multicast to users with different demands

Conclusions
A New Approach to Caching

- Main gain in caching is global
  - $\Rightarrow$ Coded multicast to users with different demands
- Global cache size matters

Conclusions

A New Approach to Caching

- Main gain in caching is global
  $\Rightarrow$ Coded multicast to users with different demands
- Global cache size matters
- Statistically identical users $\Rightarrow$ different cache content

Conclusions

A New Approach to Caching

- Main gain in caching is global
  - $\Rightarrow$ Coded multicast to users with different demands
- Global cache size matters
- Statistically identical users $\Rightarrow$ different cache content
- Significant improvement over uncoded caching schemes
  - $\Rightarrow$ Reduction in rate up to order of number of users

Conclusions

A New Approach to Caching

- Main gain in caching is global
  - ⇒ Coded multicast to users with different demands
- Global cache size matters
- Statistically identical users ⇒ different cache content
- Significant improvement over uncoded caching schemes
  - ⇒ Reduction in rate up to order of number of users
- Key open questions: block length, state, large-scale implementation

References I

Cisco, "The Zettabyte era: Trends and analysis," Tech. Rep., Jun. 2017.

M. A. Maddah-Ali and U. Niesen, "Fundamental limits of caching," *IEEE Trans. Inf. Theory*, vol. 60, no. 5, pp. 2856–2867, May 2014.

U. Niesen and M. A. Maddah-Ali, "Coded caching for delay-sensitive content," in *Proc. IEEE ICC*, Jun. 2015, pp. 5559–5564.

——, "Coded caching with nonuniform demands," *IEEE Trans. Inf. Theory*, vol. 63, pp. 1146–1158, Feb. 2017.

M. Ji, A. M. Tulino, J. Llorca, and G. Caire, "Order-optimal rate of caching and coded multicasting with random demands," *IEEE Trans. Inf. Theory*, vol. 63, pp. 3923–3949, Apr. 2017.

References II

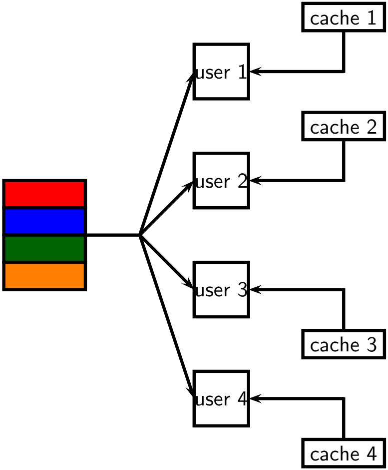
- J. Zhang, X. Lin, and X. Wang, "Coded caching under arbitrary popularity distributions," *IEEE Trans. Inf. Theory*, vol. 64, pp. 98–107, Jan. 2018.
- M. A. Maddah-Ali and U. Niesen, "Decentralized coded caching attains order-optimal memory-rate tradeoff," *IEEE/ACM Trans. Netw.*, vol. 23, pp. 1029–1040, Aug. 2015.
- J. Hachem, N. Karamchandani, and S. Diggavi, "Coded caching for multi-level popularity and access," *IEEE Trans. Inf. Theory*, vol. 63, pp. 3108–3141, May 2017.
- R. Pedarsani, M. A. Maddah-Ali, and U. Niesen, "Online coded caching," *IEEE/ACM Trans. Netw.*, vol. 24, pp. 836–845, Apr. 2016.
- N. Karamchandani, U. Niesen, M. A. Maddah-Ali, and S. Diggavi, "Hierarchical coded caching," *IEEE Trans. Inf. Theory*, vol. 62, pp. 3212–3229, Jun. 2016.

References III

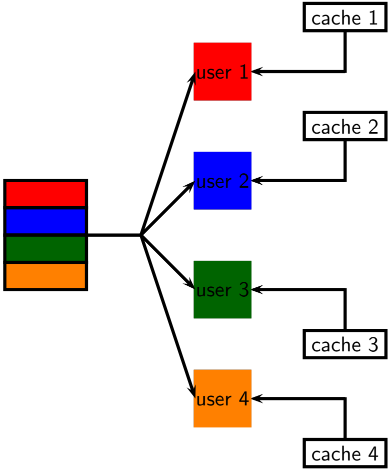
M. Ji, G. Caire, and A. F. Molisch, “Fundamental limits of caching in wireless D2D networks,” *IEEE Trans. Inf. Theory*, vol. 62, no. 2, pp. 849–869, Feb. 2016.

K. Shanmugam, A. M. Tulino, and A. G. Dimakis, “Coded caching with linear subpacketization is possible using Rusza-Szemeredi graphs,” in *Proc. IEEE ISIT*, Jun. 2017, pp. 2157–8117.

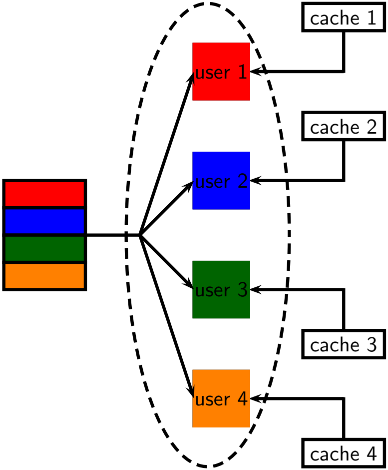
Can We Do Better?



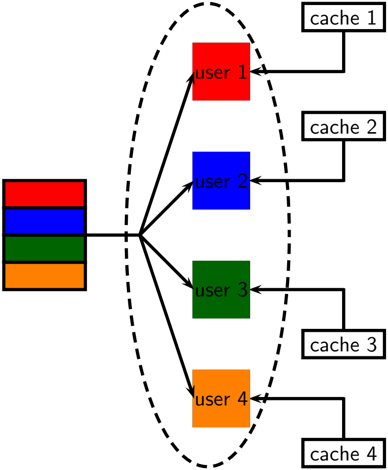
Can We Do Better?



Can We Do Better?

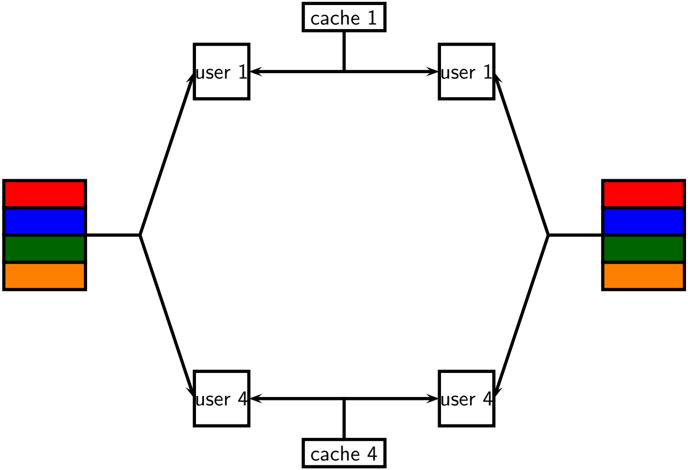


Can We Do Better?



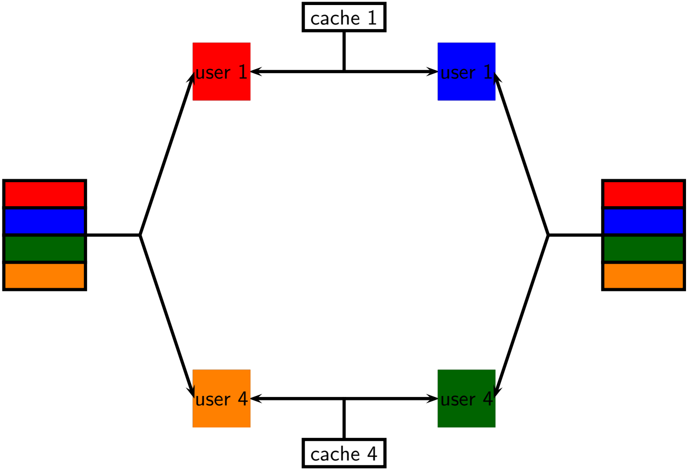
- $R + 4M \geq 4 \qquad \Rightarrow \qquad R \geq 4 - 4M$

Can We Do Better?



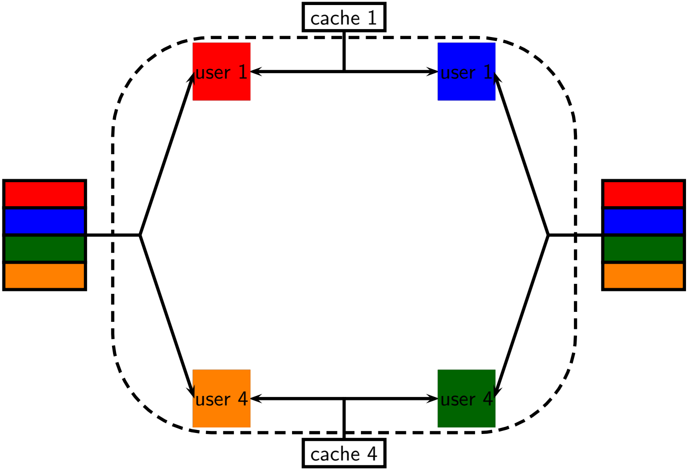
- $R + 4M \geq 4 \qquad \Rightarrow \qquad R \geq 4 - 4M$

Can We Do Better?



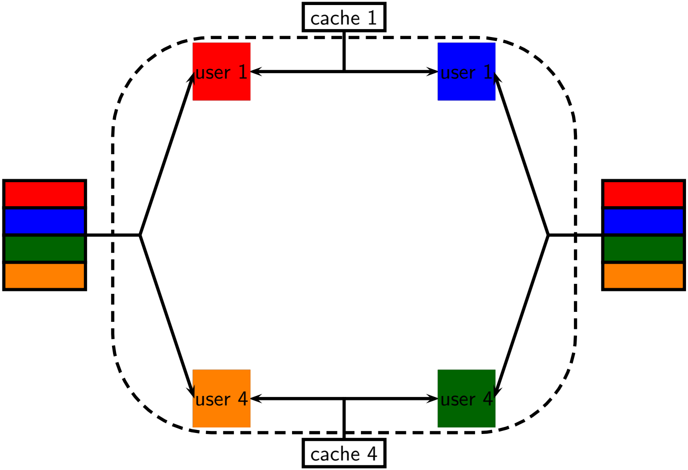
- $R + 4M \geq 4 \qquad \Rightarrow \qquad R \geq 4 - 4M$

Can We Do Better?



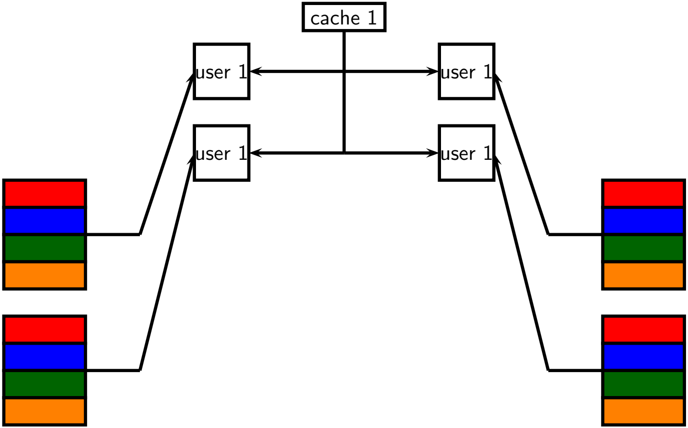
- $R + 4M \geq 4 \qquad \Rightarrow \qquad R \geq 4 - 4M$

Can We Do Better?



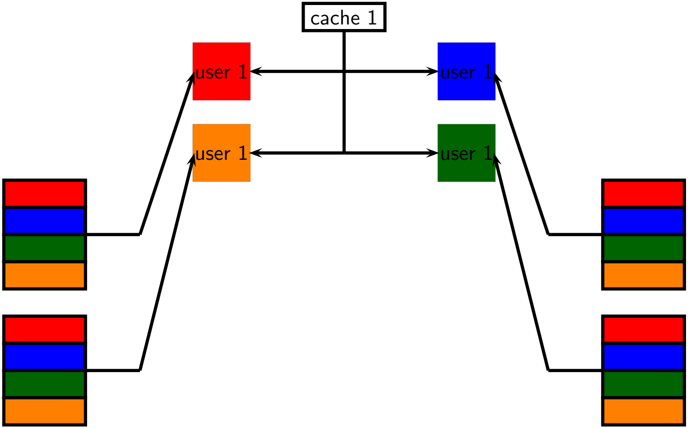
- $R + 4M \geq 4 \quad \Rightarrow \quad R \geq 4 - 4M$
- $2R + 2M \geq 4 \quad \Rightarrow \quad R \geq 2 - M$

Can We Do Better?



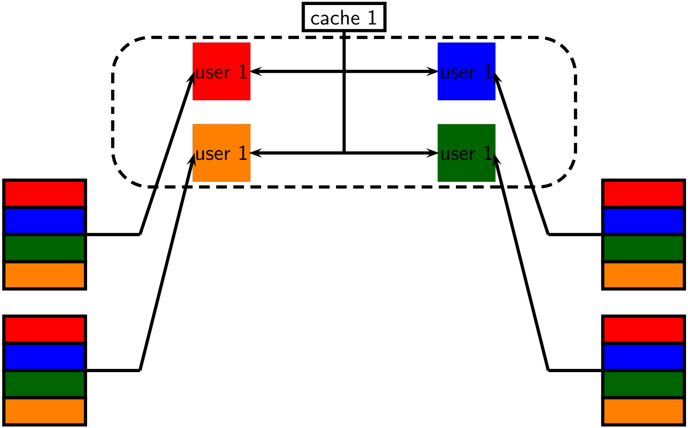
- $R + 4M \geq 4 \quad \Rightarrow \quad R \geq 4 - 4M$
- $2R + 2M \geq 4 \quad \Rightarrow \quad R \geq 2 - M$

Can We Do Better?



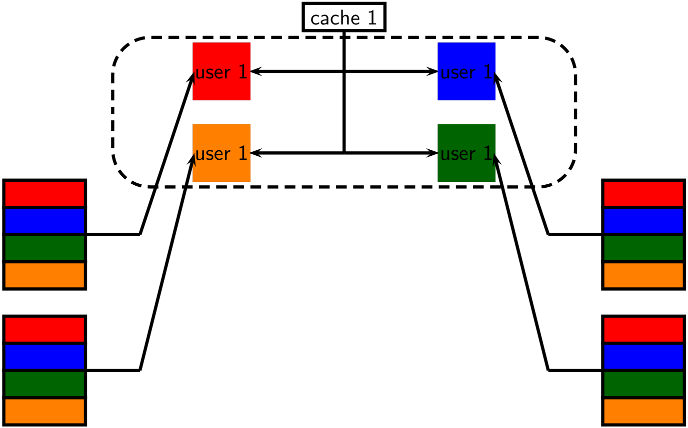
- $R + 4M \geq 4 \qquad \Rightarrow \qquad R \geq 4 - 4M$
- $2R + 2M \geq 4 \qquad \Rightarrow \qquad R \geq 2 - M$

Can We Do Better?



- $R + 4M \geq 4 \quad \Rightarrow \quad R \geq 4 - 4M$
- $2R + 2M \geq 4 \quad \Rightarrow \quad R \geq 2 - M$

Can We Do Better?

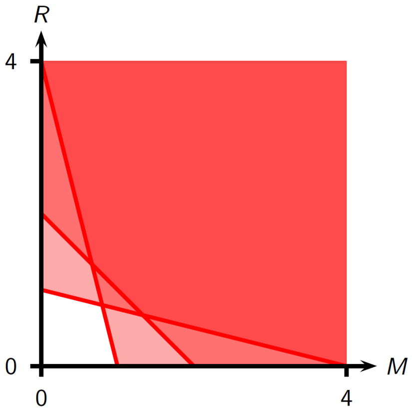


- $R + 4M \geq 4 \quad \Rightarrow \quad R \geq 4 - 4M$
- $2R + 2M \geq 4 \quad \Rightarrow \quad R \geq 2 - M$
- $4R + M \geq 4 \quad \Rightarrow \quad R \geq 1 - M/4$

Can We Do Better?

- This can be rewritten as

$$R \geq \max\{4 - 4M,\ 2 - M,\ 1 - M/4\}$$



Can We Do Better?

- This can be rewritten as

$$R \geq \max\left\{4 - 4M,\ 2 - M,\ 1 - M/4\right\}$$

- For general N and K

$$R \geq \max_s \left(s - \frac{s}{\lfloor N/s \rfloor} M \right)$$

- Comparing with achievable rate yields the theorem