Fundamental Limits of Caching

Urs Niesen
Jointly with Mohammad Maddah-Ali

Bell Labs, Alcatel-Lucent
Video on Demand

Video on demand is getting increasingly popular:

- Netflix streaming service
- Amazon Instant Video
- Hulu
- Verizon / Comcast on Demand
- ...
Video on Demand

Video on demand is getting increasingly popular:

- Netflix streaming service
- Amazon Instant Video
- Hulu
- Verizon / Comcast on Demand
- ...

⇒ Places significant stress on service provider’s networks
Video on Demand

Video on demand is getting increasingly popular:

- Netflix streaming service
- Amazon Instant Video
- Hulu
- Verizon / Comcast on Demand
- ...

⇒ Places significant stress on service provider’s networks
⇒ Caching (prefetching) can be used to mitigate this stress
Caching (Prefetching)

![Normalized demand over time of day](chart)

- The chart shows the normalized demand over the course of a day.
- The demand peaks around midday and decreases in the early evening.
- Caching strategies can be optimized based on this demand pattern.
Caching (Prefetching)

High temporal traffic variability
Caching (Prefetching)

- High temporal traffic variability
- Caching can help smooth traffic
The Role of Caching

Conventional beliefs about 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

Insights from this work:
The Role of Caching

Conventional beliefs about caching:

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

Insights from this work:

- The main gain in caching is \textit{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

Insights from this work:
- The main gain in caching is global
- Global cache size matters
The Role of Caching

Conventional beliefs about caching:
- Caches useful to deliver content \textit{locally}
- \textit{Local} cache size matters
- Statistically identical users $\Rightarrow$ identical cache content

Insights from this work:
- The main gain in caching is \textit{global}
- \textit{Global} cache size matters
- Statistically identical users $\Rightarrow$ \textit{different} cache content
Problem Setting

Server

Shared link

$K$ users

Caches
Problem Setting

server

shared link

$K$ users

caches

$N$ files, $N \geq K$ for simplicity
Problem Setting

server

shared link

$K$ users

caches

$N$ files

size $M$
Problem Setting

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

$K$ users

$caches$

$N$ files

$M$ size
Problem Setting

$K$ users

$M$ caches

$N$ shared link

Delivery:
Problem Setting

Delivery: - requests are revealed to server
Problem Setting

$K$ users caches size $M$

$N$ files

server

shared link

$K$ users

caches

size $M$

Delivery: - requests are revealed to server
- server sends arbitrary function of files
Problem Setting

- \( K \) users
- \( M \) cache size
- \( N \) files

Delivery:
- Requests are revealed to server
- Server sends arbitrary function of files
Problem Setting

$K$ users
$M$ caches
$N$ files

server
shared link

$R(M)$

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

$N$ files, $K$ users, cache size $M$
Conventional Caching Scheme

$N$ files, $K$ users, cache size $M$
Conventional Caching Scheme

N files, K users, cache size M

Diagram: A hierarchical caching structure with N levels and M/N cache spaces at each level.
Conventional Caching Scheme

$N$ files, $K$ users, cache size $M$
Conventional Caching Scheme

$N$ files, $K$ users, cache size $M$
Conventional Caching Scheme

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

Performance of conventional scheme:

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

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

Performance of conventional scheme:

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

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

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

- $R$ axis
- $M$ axis
- Dashed line: conventional scheme
Conventional Caching Scheme

\[ N = 4 \text{ files}, \quad K = 4 \text{ users} \]
Conventional Caching Scheme

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

- Conventional scheme

![Graph showing the relationship between $R$ and $M$ with $N = 8$ and $K = 8$.]
Conventional Caching Scheme

\[ N = 16 \text{ files}, \quad K = 16 \text{ users} \]
Conventional Caching Scheme

\[ N = 32 \text{ files}, \quad K = 32 \text{ users} \]
Conventional Caching Scheme

\[ N = 64 \text{ files, } K = 64 \text{ users} \]

![Diagram showing the conventional scheme with N=64 files and K=64 users. The graph illustrates the relationship between M and R, with a dashed line representing the conventional scheme.]
Conventional Caching Scheme

\[ N = 128 \text{ files, } K = 128 \text{ users} \]
Conventional Caching Scheme

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

$R$ vs $M$ graph with a dashed line indicating the conventional scheme.
Conventional Caching Scheme

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

$R$ vs $M$

- Dashed line: conventional scheme
Proposed Caching Scheme

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

Design guidelines advocated in this work:

- The main gain in caching is global
- Global cache size matters
- Different cache content at users
Proposed Caching Scheme

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

Design guidelines advocated in this work:

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

Performance of proposed scheme:

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

\[ N = 2 \text{ files,} \quad K = 2 \text{ users} \]

- Green dashed line: conventional scheme
- Blue solid line: proposed scheme

Graph showing the performance comparison between the conventional and proposed caching schemes.
Proposed Caching Scheme

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

- conventional scheme
- proposed scheme
Proposed Caching Scheme

\[ N = 8 \text{ files, } K = 8 \text{ users} \]
Proposed Caching Scheme

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

- $R$ vs. $M$
- Green dashed line: conventional scheme
- Blue solid line: proposed scheme
Proposed Caching Scheme

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

$R$ vs $M$

- **Conventional scheme**
- **Proposed scheme**
Proposed Caching Scheme

\[ N = 64 \text{ files}, \quad K = 64 \text{ users} \]
Proposed Caching Scheme

\[ N = 128 \text{ files, } K = 128 \text{ users} \]

- Green dashed line: conventional scheme
- Blue line: proposed scheme
Proposed Caching Scheme

N = 256 files, K = 256 users

- dashed line: conventional scheme
- solid line: proposed scheme
Proposed Caching Scheme

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

$R$ vs. $M$

- Conventional scheme (dashed green line)
- Proposed scheme (solid blue line)
Recall: **Conventional Scheme**

\( N = 2 \) files, \( K = 2 \) users, cache size \( M = 1 \)
Recall: **Conventional Scheme**

$N = 2$ files, $K = 2$ users, cache size $M = 1$
Recall: Conventional Scheme

$N = 2$ files, $K = 2$ users, cache size $M = 1$
Recall: Conventional Scheme

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

Diagram:

- Root: $A_1, A_2, B_1, B_2$
- Left child: $A_1, B_1$
- Right child: $A_1, B_1$
Recall: Conventional Scheme

\( N = 2 \) files, \( K = 2 \) users, cache size \( M = 1 \)
Recall: Conventional Scheme

\( N = 2 \) files, \( K = 2 \) users, cache size \( M = 1 \)
Recall: **Conventional Scheme**

\( N = 2 \) files, \( K = 2 \) users, cache size \( M = 1 \)
Recall: Conventional Scheme

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

⇒ Identical cache content at users
⇒ Gain from delivering content locally
Recall: Conventional Scheme

$N = 2$ files, $K = 2$ users, cache size $M = 1$
Recall: Conventional Scheme

$N = 2$ files, $K = 2$ users, cache size $M = 1$
Recall: **Conventional Scheme**

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

![Diagram for Conventional Scheme]

- $A_1, A_2$
- $B_1, B_2$
- $A_2, B_2$
- $A_1, B_1$
- $A_1, B_1$
Recall: Conventional Scheme

\[ N = 2 \text{ files, } K = 2 \text{ users, cache size } M = 1 \]

\[ A_1, A_2, B_1, B_2 \]

\[ A_2, B_2 \]

\[ A, B \]

\[ A_1, B_1 \]

\[ A_1, B_1 \]

⇒ Multicast only possible for users with same demand
Recall: Conventional Scheme

\( N = 2 \) files, \( K = 2 \) users, cache size \( M = 1 \)
Proposed Scheme

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

\( N = 2 \) files, \( K = 2 \) users, cache size \( M = 1 \)
Proposed Scheme

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

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

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

$A_1, A_2$
$B_1, B_2$

$A_1, B_1$
$A_2, B_2$
Proposed Scheme

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

\( N = 2 \) files, \( K = 2 \) users, cache size \( M = 1 \)
Proposed Scheme

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

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

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

\[
\begin{align*}
A_1, A_2 \\
B_1, B_2
\end{align*}
\]

$A_2 \oplus B_1$

$A$

$B$

\[
\begin{align*}
A_1, B_1 \\
A_2, B_2
\end{align*}
\]

$⇒$ Different cache content at users

$⇒$ Multicast to 2 users with different demands
Proposed Scheme

\( N = 2 \) files, \( K = 2 \) users, cache size \( M = 1 \)

\[ A_1, A_2 \]
\[ B_1, B_2 \]

\[ B_2 \oplus A_1 \]

\[ A \]
\[ A_1, B_1 \]
\[ A_2, B_2 \]

\[ B_2 \oplus B_1 \]

\[ A \]
\[ A_1, B_1 \]
\[ A_2, B_2 \]

\[ B \]
\[ A_1, B_1 \]
\[ A_2, B_2 \]

\[ \Rightarrow \] Works for all possible user requests

\[ \Rightarrow \] Simultaneous multicasting gain
Proposed Scheme

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

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

- Scheme can be generalized to arbitrary:
  - Number of files $N$
  - Number of users $K$
  - Cache size $M$
Proposed Scheme

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

- Scheme can be generalized to arbitrary:
  - Number of files $N$
  - Number of users $K$
  - Cache size $M$

- Enables multicast to $KM/N + 1$ users with different demands
Comparison of the Two Schemes

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

- Conventional scheme: $R(M) = K \cdot (1 - M/N)$
- Proposed 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

- Conventional scheme: \( R(M) = K \cdot (1 - M/N) \)
- Proposed 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 $M$

- Conventional scheme: $R(M) = K \cdot (1 - M/N)$
- Proposed 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 \(M\)

- Conventional scheme: \(R(M) = K \cdot (1 - M/N)\)
- Proposed 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 $M$

- Conventional scheme: $R(M) = K \cdot (1 - M/N)$
- Proposed 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
Can We Do Better?

Theorem

*The proposed scheme is *optimal* to within a *constant* factor in rate.*
Can We Do Better?

Theorem

The proposed scheme is optimal to within a constant factor in rate.

⇒ Information-theoretic bound
Can We Do Better?

Theorem

The proposed scheme is optimal to within a constant factor in rate.

⇒ Information-theoretic bound

⇒ Constant is independent of problem parameters $N, K, M$
The proposed scheme is optimal to within a constant factor in rate.

⇒ Information-theoretic bound

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

⇒ No other significant gain besides local and global
Conclusions
A New Approach to Caching
Conclusions
A New Approach to Caching

- The main gain in caching is global
  - Multicast to users with different demands
The main gain in caching is global
  ⇒ Multicast to users with different demands

Global cache size matters
Conclusions
A New Approach to Caching

- The main gain in caching is global
  ⇒ Multicast to users with different demands

- **Global** cache size matters

- Statistically identical users ⇒ different cache content
Conclusions
A New Approach to Caching

- The main gain in caching is global
  \[\Rightarrow\] Multicast to users with different demands

- Global cache size matters

- Statistically identical users \[\Rightarrow\] different cache content

- Significant improvement over conventional caching schemes
  \[\Rightarrow\] Reduction in rate up to order of number of users
Conclusions
A New Approach to Caching

- The main gain in caching is global
  - Multicast to users with different demands

- Global cache size matters

- Statistically identical users ⇒ different cache content

- Significant improvement over conventional caching schemes
  - Reduction in rate up to order of number of users

- Papers available on arXiv
  - Niesen, Maddah-Ali: “Coded Caching with Nonuniform Demands”