Contents on the Move
Content Caching and Delivery at the Wireless Network Edge

Deniz Gündüz
Imperial College London

25 April 2018
Queen Mary University of London
London, UK
Video demand dominates traffic (78% by 2021)

75% of Facebook video browsing, 40% of Netflix downloads performed on smartphones

We need a content aware network design

Asymmetric resource usage

Delay-tolerant, asynchronous access

Most traffic due to a few viral/ popular video files

Demand and access patterns highly predictable

Storage is relatively cheap, while bandwidth is extremely expensive!
Video demand dominates traffic (78% by 2021)

75% of Facebook video browsing, 40% of Netflix downloads performed on smartphones

We need a content aware network design

- Asymmetric resource usage
- Delay-tolerant, asynchronous access
- Most traffic due to a few viral/popular video files
- Demand and access patterns highly predictable

Storage is relatively cheap, while bandwidth is extremely expensive!
Video demand dominates traffic (78% by 2021)

75% of Facebook video browsing, 40% of Netflix downloads performed on smartphones

We need a content aware network design

Asymmetric resource usage

Delay-tolerant, asynchronous access

Most traffic due to a few viral/ popular video files

Demand and access patterns highly predictable

Storage is relatively cheap, while bandwidth is extremely expensive!
Video demand dominates traffic (78% by 2021)
75% of Facebook video browsing, 40% of Netflix downloads performed on smartphones
We need a content aware network design

- Asymmetric resource usage
- Delay-tolerant, asynchronous access
- Most traffic due to a few viral/popular video files
- Demand and access patterns highly predictable

Storage is relatively cheap, while bandwidth is extremely expensive!
Content provider (e.g. Netflix, BBC, Facebook) contracts with a CDN (e.g. Akamai, LimeLight)

- Balance traffic, reduce latency, ...
- This is in the core network
Content provider (e.g. Netflix, BBC, Facebook) contracts with a CDN (e.g. Akamai, LimeLight)

- Balance traffic, reduce latency, ...
- This is in the core network
- Bring content to the edge (e.g., Netflix Open Connect)
Two-phase protocol:
  - **Placement phase**: off-peak hours, user demands unknown
  - **Delivery phase**: peak hours, demands revealed

Library of $N$ files, each consisting of $F$ bits

$K$ users, each equipped with a cache of size $M$

Each user requests one file

**Error-free shared delivery link**: Satisfy all demands simultaneously

What is the minimum number of bits that must be delivered sufficient to satisfy all demand combinations?

What is the trade-off between cache capacity and number of delivered bits?

---

Coded Proactive Content Caching

Two-phase protocol:
- **Placement phase**: off-peak hours, user demands unknown
- **Delivery phase**: peak hours, demands revealed

Library of $N$ files, each consisting of $F$ bits
$K$ users, each equipped with a cache of size $M$
Each user requests one file
**Error-free shared delivery link**: Satisfy all demands simultaneously

What is the minimum number of bits that must be delivered sufficient to satisfy all demand combinations?
What is the trade-off between cache capacity and number of delivered bits?

---

Coded Proactive Content Caching

Two-phase protocol:
- **Placement phase**: off-peak hours, user demands unknown
- **Delivery phase**: peak hours, demands revealed

Library of $N$ files, each consisting of $F$ bits
- $K$ users, each equipped with a cache of size $M$
- Each user requests one file
- **Error-free shared delivery link**: Satisfy all demands simultaneously

What is the minimum number of bits that must be delivered sufficient to satisfy all demand combinations?
What is the trade-off between cache capacity and number of delivered bits?

---

Example 1

- $N = 3$ files
- $K = 3$ users
- Cache capacity: $M = 1$
- Split each file into 3 non-overlapping equal-size subfiles:

<table>
<thead>
<tr>
<th></th>
<th>$W_1$</th>
<th>$W_2$</th>
<th>$W_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

- Cache contents after placement phase:

  ![Cache Diagram]
Delivery phase:

Delivery rate: $R_{\text{MAN}}(1) = 1$
Example 2

- \( N = 3 \) files
- \( K = 3 \) users
- Cache capacity: \( M = 2 \)
- Split each file into 3 non-overlapping equal-size subfiles:

  \[
  \begin{align*}
  W_1 & : 12 \quad 13 \quad 23 \\
  W_2 & : 12 \quad 13 \quad 23 \\
  W_3 & : 12 \quad 13 \quad 23 
  \end{align*}
  \]

- Cache contents after placement phase:

  User 1:
  - 12 13 12 13 12 13

  User 2:
  - 12 23 12 23 12 23

  User 3:
  - 13 23 13 23 13 23
Delivery phase:

\[ R_{MAN}(2) = \frac{1}{3} \]
Many improvements and variations since then...


Devices have different resolution/processing capabilities
They may request the same file, but at different resolutions
$D_k$: distortion requirement of user $k$. Without loss of generality, let

$$D_1 \geq D_2 \geq \cdots \geq D_K$$

Devices have distinct cache capacities: $M_k$

Q. Yang and D. Gündüz, **Coded caching and content delivery with heterogeneous distortion requirements**, to appear, IEEE Trans. on Information Theory.
### Scalable Coded Caching

Compress video into multiple quality layers; e.g., **scalable video coding (SVC)** in H264/ MPEG

- First layer: \( r_1 \) bits/sample
- \( k \)-th layer: \( r_k - r_{k-1} \) bits/sample
- User \( k \) wants \( D_k \) → needs first \( k \) layers
Centralized Lossy Coded Caching ($N = K = 2$)

Given $(r_1, r_2)$, five cases depending on cache capacities $M_1$ and $M_2$:

- **Case i&ii**: coded placement
- **Case iii&iv**: coded placement and coded delivery
- **Case v**: uncoded caching

Proposed layered caching scheme is optimal.

It requires coded caching and delivery simultaneously.
Two subproblems:

- Cache allocation among different layers
- Lossless caching/delivery of each layer with heterogeneous cache sizes

Cache capacity allocation:

- **Proportional Cache Allocation (PCA)**
  - Allocate cache capacity proportionally (to sizes) among requested layers

- **Ordered Cache Allocation (OCA)**
  - Allocate cache capacity to “more important” first layers
Two subproblems:
- Cache allocation among different layers
- Lossless caching/delivery of each layer with heterogeneous cache sizes

Cache capacity allocation:
- **Proportional Cache Allocation (PCA)**
  - Allocate cache capacity proportionally (to sizes) among requested layers
- **Ordered Cache Allocation (OCA)**
  - Allocate cache capacity to “more important” first layers
Identical Cache Capacities

- $D_1 \geq D_2 \geq \cdots \geq D_{10}$: $r_k = k$, $k = 1, \ldots, 10$;
- Identical cache capacities, $M_k = M$. 

![Graph with legends](N=10,K=10,M_k=M)

- Uncoded Caching
- Coded Caching with PCA
- Coded Caching with OCA
- Cut-set Bound

**Cache Size, $M$**

**Delivery Rate, $R$**
Heterogeneous Cache Capacities

- $D_1 \geq D_2 \geq \cdots \geq D_{10}: r_k = k, k = 1, \ldots, 10$;
- Heterogeneous cache capacities, $M_k = 0.2kM$.

![Graph showing delivery rate vs cache size with different caching methods: Uncoded Caching, Coded Caching with PCA, Coded Caching with OCA, and Cut-set Bound. The graph has $N = 10$, $K = 10$, and $M_k = 0.2kM$. The x-axis represents cache size, $M$, ranging from 0 to 5, and the y-axis represents delivery rate, $R$, ranging from 55 to 15.]
\[
\delta_k = \begin{cases} 
\delta_w & \text{if } k \in [K_w] \\
\delta_s & \text{if } k \in [K_w + 1 : K] 
\end{cases}
\]

Library of $N$ files: $W_1, \ldots, W_N$

Each file is distributed uniformly over $[2^{nR}] \triangleq \{1, \ldots, 2^{nR}\}$

Packet erasure broadcast channel

$$P (Y_k = y_k | X = x) = \begin{cases} 1 - \delta_k, & \text{if } y_k = x, \\ \delta_k, & \text{if } y_k = \Delta \end{cases}$$

$$P_e \triangleq \max_{(d_1, \ldots, d_K) \in [N]^K} \Pr \left\{ \bigcup_{k=1}^{K} \left\{ \hat{W}_{d_k} \neq W_{d_k} \right\} \right\}$$

$(M, R)$ is achievable, if for every $\varepsilon > 0$, $\exists n$ large enough, s.t. $P_e < \varepsilon$

$$C \triangleq \sup \{R : (M, R) \text{ is achievable}\}$$

Cache capacity of $M$ only at weak receivers
Main Result: Achievable Rate-Memory Pairs

Memory-rate pairs \((M_{(p,q)}, R_{(p,q)})\) are achievable for any \(p \in [0: K_w]\) and \(q \in [p: K_w]\):

\[
R_{(p,q)} \triangleq \frac{F \sum_{i=p}^{q} (\gamma (p, i))}{1 - \delta_w \sum_{i=p}^{q} \left( \frac{K_w - i}{i+1} \gamma (p, i) \right) + \frac{K_s}{1 - \delta_s}},
\]

\[
M_{(p,q)} \triangleq \frac{N \sum_{i=p}^{q} i \gamma (p, i)}{K_w \sum_{i=p}^{q} \gamma (p, i)} R_{(p,q)},
\]

where

\[
\gamma (p, i) \triangleq \binom{K_w}{i} \binom{K_w}{p} K_s^{-p} \left( \frac{1 - \delta_s}{1 - \delta_w} - 1 \right)^{i-p}, \text{ for } i = p, \ldots, q.
\]

---

Successive Joint Cache-Channel Coding (SCC) Scheme

- $K_w = 3$ weak RXs
- $K_s = 2$ strong RXs
- $p = 0$, $q = 2$

Rate of $W_i^{(k)}$ is $R^{(k)}$, $k = 0, 1, 2$

$R^{(0)} + R^{(1)} + R^{(2)} = R$
Successive Joint Cache-Channel Coding (SCC) Scheme

Placement phase:

- User 1
  - 1
  - 12
  - 13

- User 2
  - 2
  - 12
  - 23

- User 3
  - 3
  - 13
  - 23

Cache capacity: $M = \frac{R^{(1)}}{3} + \frac{2R^{(2)}}{3}$
- $q - p + 2 = 4$ distinct messages delivered by **time division** multiplexing
- Codewords of $i$-th message are of length $\beta_i n$ channel uses, $i = 1, \ldots, 4$:

$$\sum_{i=1}^{4} \beta_i = 1$$
Message 1:

Correct decoding if

$$\frac{R^{(2)}/3}{(1 - \delta_w)F} \leq \beta_1$$
Message 2, Part 1:

Correct decoding if

\[
\max \left\{ \frac{R^{(1)} / 3}{(1 - \delta_w)F}, \frac{R^{(1)} / 3 + 2R^{(2)} / 3}{(1 - \delta_s)F} \right\} \leq \beta_{2,1}
\]
Message 2, Part 2:

Correct decoding if

\[
\max \left\{ \frac{R^{(1)}/3}{(1 - \delta_w)F}, \frac{R^{(1)}/3 + 2R^{(2)}/3}{(1 - \delta_s)F} \right\} \leq \beta_{2,2}
\]
Correct decoding if

\[
\max \left\{ \frac{R^{(1)}/3}{(1 - \delta_w) F}, \frac{R^{(1)}/3 + 2R^{(2)}/3}{(1 - \delta_s) F} \right\} \leq \beta_{2,3}
\]

Equivalently:

\[
\max \left\{ \frac{R^{(1)}}{(1 - \delta_w) F}, \frac{R^{(1)} + 2R^{(2)}}{(1 - \delta_s) F} \right\} \leq \beta_2
\]
Message 3, Part 1:

Correct decoding if

\[
\max \left\{ \frac{R^{(0)}}{(1 - \delta_w) F}, \frac{R^{(0)} + 2R^{(1)}/3}{(1 - \delta_s) F} \right\} \leq \beta_{3,1}
\]
Message 3, Part 1:

Correct decoding if

$$\max \left\{ \frac{R^{(0)}}{(1 - \delta_w)F}, \frac{R^{(0)} + 2R^{(1)}/3}{(1 - \delta_s)F} \right\} \leq \beta_{3,2}$$
Correct decoding if

\[
\max \left\{ \frac{R^{(0)}}{1 - \delta_w} F, \frac{R^{(0)} + 2R^{(1)}/3}{(1 - \delta_s)F} \right\} \leq \beta_{3,3}
\]

Equivalently:

\[
\max \left\{ \frac{3R^{(0)}}{(1 - \delta_w)F}, \frac{3R^{(0)} + 2R^{(1)}}{(1 - \delta_s)F} \right\} \leq \beta_3
\]
Message 4:

\[
\frac{2R^{(0)}}{(1 - \delta_s) F} \leq \beta_4
\]
Achievable Memory-Rate Pair Analysis

Message 1: \( \frac{R^{(2)}/3}{(1-\delta_w)F} \leq \beta_1 \)

Message 2: \( \max \left\{ \frac{R^{(1)}}{(1-\delta_w)F}, \frac{R^{(1)} + 2R^{(2)}}{(1-\delta_s)F} \right\} \leq \beta_2 \)

Message 3: \( \max \left\{ \frac{3R^{(0)}}{(1-\delta_w)F}, \frac{3R^{(0)} + 2R^{(1)}}{(1-\delta_s)F} \right\} \leq \beta_3 \)

Message 4: \( \frac{2R^{(0)}}{(1-\delta_s)F} \leq \beta_4 \)

\( \beta_i \)s chosen such that:

\[
\frac{R^{(2)}/3}{(1-\delta_w)F} + \frac{R^{(1)}}{(1-\delta_w)F} + \frac{3R^{(0)}}{(1-\delta_w)F} + \frac{2R^{(0)}}{(1-\delta_s)F} = 1
\]

Choose rates s.t. max achieved by equality.

\[ R^{(0)} + R^{(1)} + R^{(2)} = R \]

Required cache capacity:

\[ M = \frac{R^{(1)}}{3} + \frac{2R^{(2)}}{3} \]
Summary of SCC Scheme

$W_d$  $W_d^{(p)}$  $W_d^{(i-1)}$  $W_d^{(i)}$  $W_d^{(q)}$

$coded messages$

$\left( K_w \right)_{i}$

$\delta_w$

$\delta_w$

$\delta_s$

$\delta_s$

$\delta_w$

$\delta_w$

$\delta_s$

$\delta_s$

uncoded messages
Rate-Memory Trade-off

\[ N = 50 \text{ files} \]
\[ K_w = 7 \text{ weak Rxs, } K_s = 10 \text{ strong Rxs} \]
\[ F = 20, \delta_w = 0.8, \delta_s = 0.2 \]
$N = 100$ files
$K_w = 20$ weak Rxs, $K_s = 10$ strong Rxs
$F = 50$, $\delta_s = 0.2$

<table>
<thead>
<tr>
<th>Cache size, $M$</th>
<th>Rate, $R$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>50</td>
<td>0.5</td>
</tr>
<tr>
<td>100</td>
<td>1</td>
</tr>
<tr>
<td>150</td>
<td>1.5</td>
</tr>
<tr>
<td>200</td>
<td>2</td>
</tr>
<tr>
<td>250</td>
<td>2.5</td>
</tr>
<tr>
<td>300</td>
<td>3</td>
</tr>
<tr>
<td>350</td>
<td>3.5</td>
</tr>
<tr>
<td>400</td>
<td>4</td>
</tr>
</tbody>
</table>

Deniz Gündüz (Imperial College)
System overview

- $K_T \times K_R$ interference channel
- Transmitter cache: $M_T F$
- Receiver cache $M_R F$

Sum Degrees-of-Freedom

\[
\text{DoF}(M_T, M_R) = \liminf_{P \to \infty} \frac{C(M_T, M_R, P)}{\log(P)}
\]

- Decentralized caching at user terminals (RXs)

Novel scheme combining:

- Zero-forcing
- Interference cancellation
- Interference alignment
Fog-Aided Radio Access Networks

System overview

- Fronthaul connections to base stations
- Uncached contents can be delivered from the cloud server

Normalized Delivery Time

\[ \delta(M_T, M_R) = \lim_{P \to \infty} \lim_{F \to \infty} \frac{T_F + T_E}{F / \log(P)}. \]

- Orthogonal backhaul links
- Fronthaul capacity \( r \) unknown during placement
- Serial/pipelined fronthaul delivery
- Hard-transfer fronthauling
- Joint edge and cloud delivery


Channel and network conditions vary over time

- State of the art: Reactive content delivery
- User behaviour (demands and mobility) are highly predictable
- **Contents can be pushed in advance when channel is good.**

---


Demands known/predicted in advance

Finite capacity cache at user terminal

System model:
- Duration of time slot $i$: $\tau_i$
- User demand rate: $d_i$
- Channel state: $h_i$
- Cache capacity: $B$
- Rate-power function:
  \[ r(t) = \log(1 + h(t)p(t)) \]

Objective: Minimize energy consumption over $N$ timeslots:

\[
\min_{i \geq 0} \sum_{i=1}^{N} \tau_i \frac{e^{r_i} - 1}{h_i}
\]

s.t. \[ \sum_{i=1}^{n} \tau_i (d_i - r_i) \leq 0, \text{ for } n = 1, \ldots, N, \]

\[ \sum_{i=1}^{n} \tau_i (r_i - d_i) - B \leq 0, \text{ for } n = 1, \ldots, N. \]
- Download demands over a longer period, and in better channel conditions
- Each file can be downloaded only in advance, not later than when it is requested
- Proactive caching amount is limited by cache memory
Proactive Caching in a Dynamic Environment

- Contents generated randomly, with random lifetime
- User accesses at random time instants to download all relevant contents (e.g., online social network)
- Cost = Channel cost of download × downloaded data
- **Goal:** Minimize long-term average cost
- Proactively cache content at favourable channel conditions

---

System State:

- Relevant contents outside cache ⇒ $O_t$.
- Contents inside cache ⇒ $I_t$ ($|I_t| \leq B$).
- Elapsed time since last user access ⇒ $E_t$.
- Energy cost of downloading a content ⇒ $C_t$ ($0 < C_t \leq C_{max}$): i.i.d. over time.
Markov decision process with side information (MDP-SI).

- **State** ($s \in S$):
  - **Controllable state**: $(O_t, I_t, E_t)$.
  - **Uncontrollable state**: $C_t \Rightarrow$ side information

- **Action** ($a \in A_s$): $A_t = (A_t^{(1)}, A_t^{(2)})$.
- **Transition probability**: $P(S_{t+1}|S_t, A_t)$.
- **Cost function**: $\mu(S_t, A_t) = C_t \cdot |A_t^{(1)}|$.
- **Objective function**: $\rho = \lim_{T \to \infty} \mathbb{E} \left[ \frac{1}{T} \sum_{t=1}^{T} \mu(S_t, A_t) \right]$. 
Markov decision process with side information (MDP-SI).

- **State** ($s \in S$):
  - Controllable state: $(O_t, I_t, E_t)$.
  - Uncontrollable state: $C_t \Rightarrow$ side information
- **Action** ($a \in A_s$): $A_t = (A_t^{(1)}, A_t^{(2)})$.
- **Transition probability**: $P(S_{t+1}|S_t, A_t)$.
- **Cost function**: $\mu(S_t, A_t) = C_t \cdot |A_t^{(1)}|$.
- **Objective function**: $\rho = \lim_{T \to \infty} \mathbb{E} \left[ \frac{1}{T} \sum_{t=1}^{T} \mu(S_t, A_t) \right]$. 
For any state $s = (O, I, E) \in S$, the optimal policy $\pi^*(s)$ has a threshold structure with respect to cost $C$.

Let

- $l_1 \leq \cdots \leq l_B$ : contents in the cache ($I$).
- $L_1 \geq \cdots \geq L_B$ : $B$ contents out of cache ($O$) with highest lifetimes.

There exists $B' \leq B$ and corresponding threshold values:

$$T(a_{B'}) \leq T(a_{B' - 1}) \leq \cdots \leq T(a_1) \leq C_{\text{max}},$$

and the optimal policy performs simple actions $a_i = (l_i | L_i)$, if $C \leq T(a_i)$ and $E > 0$. 
For any state $s = (\mathcal{O}, \mathcal{I}, E) \in \mathcal{S}$, the optimal policy $\pi^*(s)$ has a threshold structure with respect to cost $C$.

Let

- $l_1 \leq \cdots \leq l_B$ : contents in the cache ($\mathcal{I}$).
- $L_1 \geq \cdots \geq L_B$ : $B$ contents out of cache ($\mathcal{O}$) with highest lifetimes.

Exist $B' \leq B$ and corresponding threshold values:

$$T(a_{B'}) \leq T(a_{B'-1}) \leq \cdots \leq T(a_1) \leq C_{\text{max}},$$

and the optimal policy performs simple actions $a_i = (l_i | L_i)$, if $C \leq T(a_i)$ and $E > 0$. 
**Longest lifetime in–Shortest lifetime out:**

- Swap largest $L \in \mathcal{O}$ with the smallest $l \in \mathcal{I}$, if $C_t \leq T(a)_{a=(l|L)}$, until no more swaps can be performed.
- Single threshold value for each pair $(l|L)$ of lifetimes.
- Parametrized by threshold values: $\theta = T(l|L)$ for all $L > l$. 
Threshold values obtained using linear function approximation (LFA) as

\[ T(a)_{a=(l|L)} = \sum_{i=0}^{K_{max}} \phi(i)\theta_i(l, L) = \Phi^\top \theta(l, L), \]

\(K_{max}: \) maximum lifetime

\(\Phi_t = [\phi_t(0), \phi_t(1), \ldots, \phi_t(K_{max})]: \text{frequency vector}\)

\[ \phi(i) \triangleq \frac{\sum_{l \in C} \mathbb{I}_{\{l=i\}}}{B}, \quad \text{for} \quad i = 0, 1, \ldots, K_{max}, \]

\(\theta_i(l, L): \) coefficients to be optimized for each simple action.
A model free policy search technique using stochastic gradient descent.

**Policy Gradient Algorithm**

- generate “samples” with $P(s'|s, a)$ and the probability density function $f_C(c)$
  - Results in trajectory $\tau_{\pi_\theta} = (S_1, C_1, A_1), \ldots , (S_T, C_T, A_T)$ i.e., $\tau_{\pi_\theta}, T \sim P_{\theta,T}(\tau_{\pi_\theta}) = P(\tau_{\pi_\theta}, T|\theta)$.
- Evaluate average sample cost $J_{\pi_\theta} = \frac{1}{T} \sum_{t=1}^{T} \mu(S_t, A_t)$
- Update $\theta$ in the direction that decreases $\rho^{\pi_\theta} = \mathbb{E}[J_{\pi_\theta}]:$
  $$\theta_{j+1} = \theta_j - \lambda \nabla_{\theta} \rho^{\pi_\theta},$$
  where $\lambda > 0$ is the step size, $j$ is the current iteration step and
  $$\nabla_{\theta} \rho^{\pi_\theta} = \int_{\tau} \nabla_{\theta} P_{\theta}(\tau_{\pi_\theta}) J_{\pi_\theta} d\tau.$$
Performance Bounds

- **Unlimited cache capacity (LB-UC)**
  - Decouples actions for contents, $A_t^{(2)} = \emptyset$, $\forall t$
  - Threshold $T_L$: Content with lifetime $L$ is downloaded if $C \leq T_L$.
    \[ 0 \leq T_1 \leq \cdots \leq T_{K_{max}} \leq C_{max} \]
  - Threshold obtained using value iteration algorithm (VIA)

- **Non-causal knowledge of user access times (LB-NCK)**
  - For any time-to-user access $t'$, contents are downloaded if $C_t \leq T_{t'}$.
    \[ 0 \leq T_{D_{max}} \leq \cdots \leq T_1 \leq C_{max} \]
  - where $D_{max}$ is the bound on the user access interval.
  - Threshold values obtained using VIA.
Performance Bounds

- **Unlimited cache capacity (LB-UC)**
  - Decouples actions for contents, $A_t^{(2)} = \emptyset$, $\forall t$
  - Threshold $T_L$: Content with lifetime $L$ is downloaded if $C \leq T_L$.
    \[
    0 \leq T_1 \leq \cdots \leq T_{K_{max}} \leq C_{max}
    \]
  - Threshold obtained using value iteration algorithm (VIA)

- **Non-causal knowledge of user access times (LB-NCK)**
  - For any time-to-user access $t'$, contents are downloaded if $C_t \leq T_{t'}$.
    \[
    0 \leq T_{D_{max}} \leq \cdots \leq T_1 \leq C_{max}
    \]
  - where $D_{max}$ is the bound on the user access interval.
  - Threshold values obtained using VIA.
Percentage Improvement over LISO with FDM:

- LFA with LRM → up to 5.6%.
- LFA with FDM → up to 4.4%.
- LISO with LRM → up to 4.2%.
Random mobility patterns

- Maximum distance separable (MDS) coded content storage
- How to allocate cached to contents with different popularities?

---


Each user connects to $\rho$ out of $P$ servers
Each server can cache $N/\rho$ files
Both coded caching and MDS coded storage need to be utilised

---

Interactive multiview streaming

How to optimally cache and deliver multiview video content to improve the free viewpoint streaming experience?

Thank You for Your Attention!