Distributed Wireless Video Caching Placement for Dynamic Adaptive Streaming

Chenglin Li*, Pascal Frossard (EPFL)
Hongkai Xiong (SJTU)
Junni Zou (SHU)

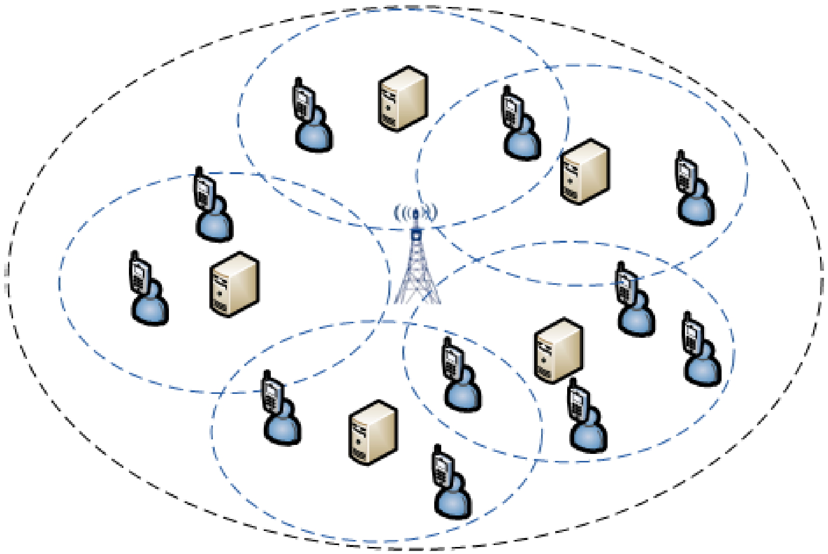
13.05.2016

Outline

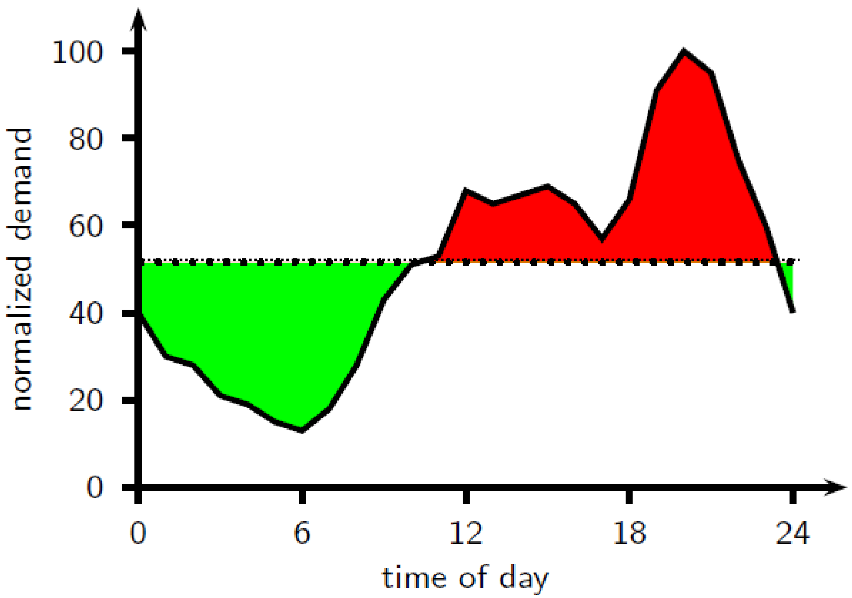
- 1. Background
- 2. Motivation and Problem Formulation
- 3. Submodularity and Approximation Algorithm
- 4. Experimental Evaluation
- 5. Conclusion and Future Direction

Why caching (pre-fetching)?

Mobile video delivery network



Video demand



Two phases:

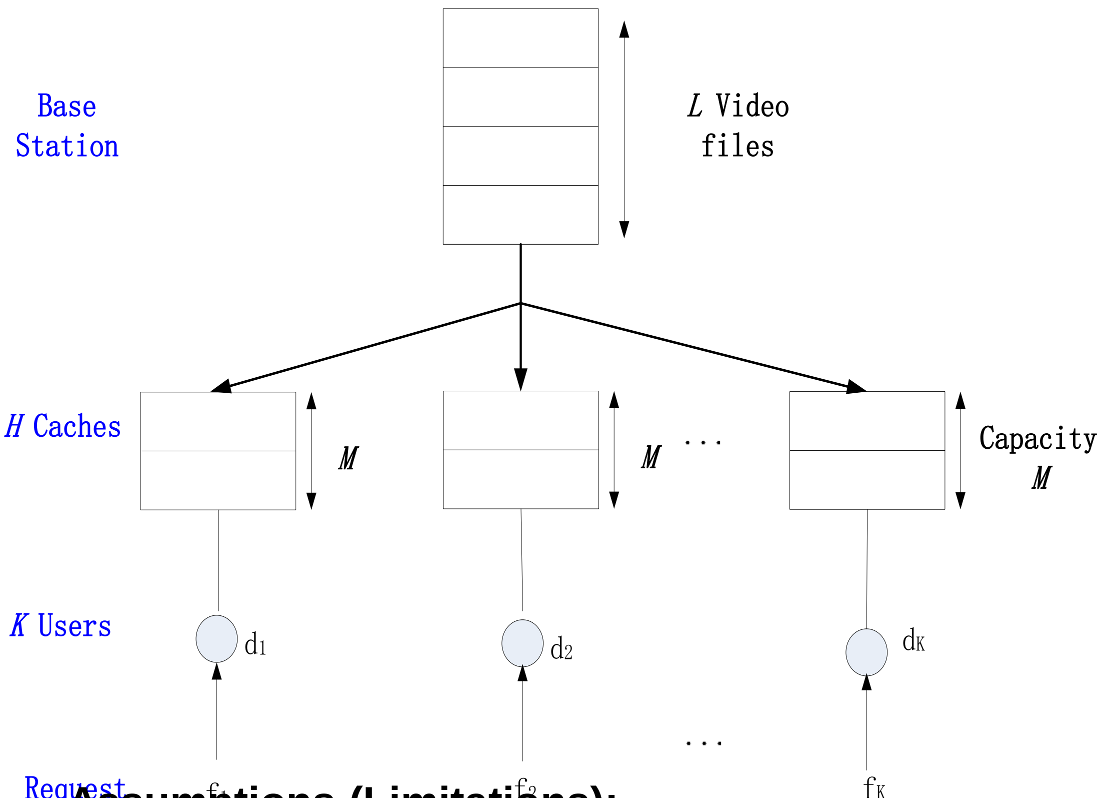
- Placement phase (e.g., 6am): populate caches
- Delivery phase (e.g. 9pm): deliver video content upon request

Address two problems:

⇒ Stress on service provider's networks

⇒ High temporal traffic variability

Ideal caching case – Some theoretical bounds



Assumptions (Limitations):

- Same video file size
- Same cache capacity
- One user only accesses to one cache

Base Station Trans. Rate

- No caching

$$R = K$$

- Caching, No Coding

$$R = K \cdot (1 - \frac{M}{L})$$

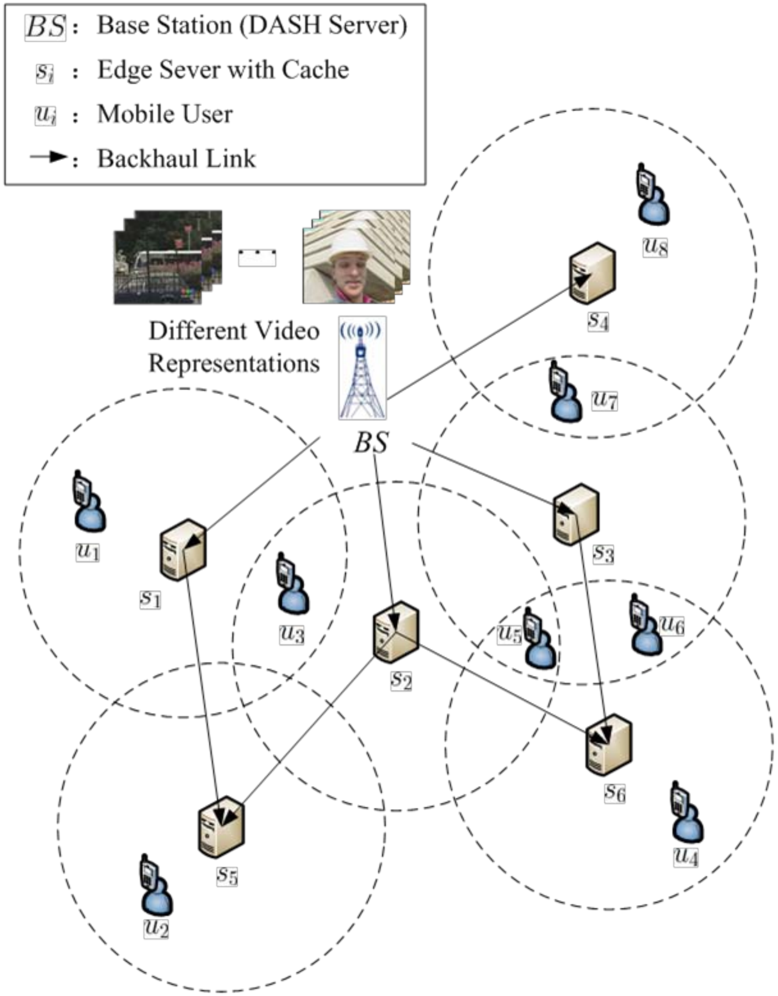
Local caching gain

- Coded Caching

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

Global caching gain

Practical DASH cache problem



Backhaul link << Local link

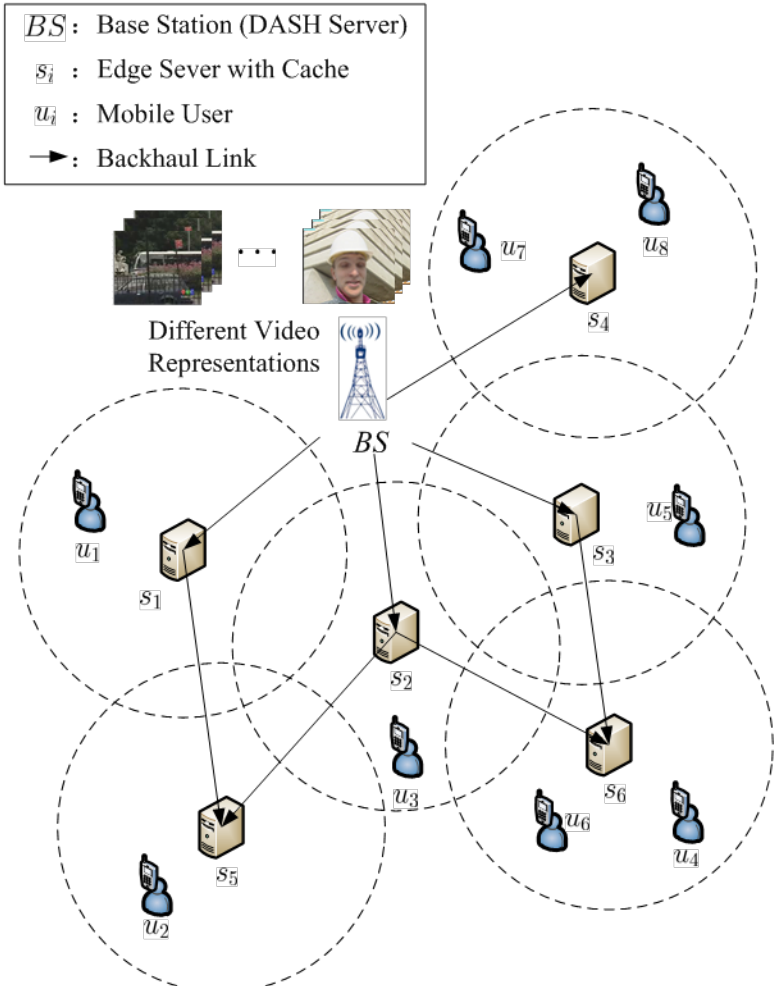
Practical issues:

- Multiple versions for a video
- Different d-R-D behavior
- Different edge serer cache capacity
- One user can access to multiple caches

Problem description:

- Given :
  - Representation set of source video files
  - File popularity distribution
  - Network topology
  - Edge sever cache capacity
  - Download delay requirement of users
- How to place representations in edge severs → total system utility maximized

Simple case
– One user to one edge sever



Trivial Case

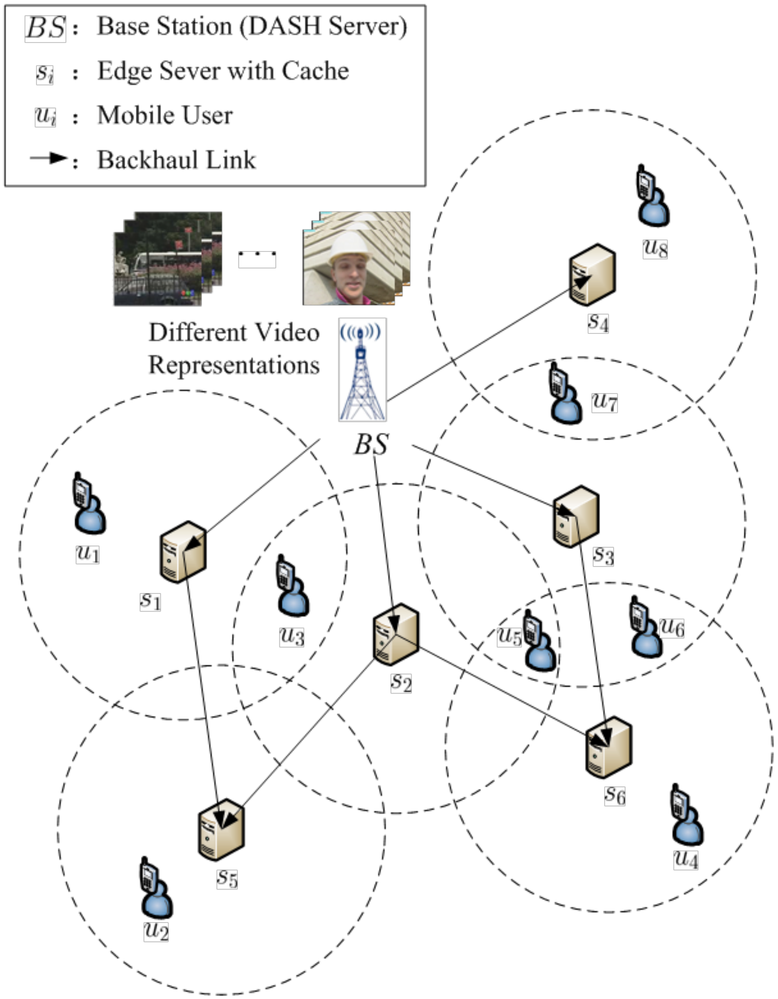
If assume:

- Each video file has the same size
- No R-D consideration
- No multiple representations for each video file

Simple strategy:

- For each edge sever, cache as many most popular video files as possible

Nontrivial case
– One user to multi-edge severs



Dense edge sever deployment

Solved for Optimal placement:

- Uncoded case
- Coded case

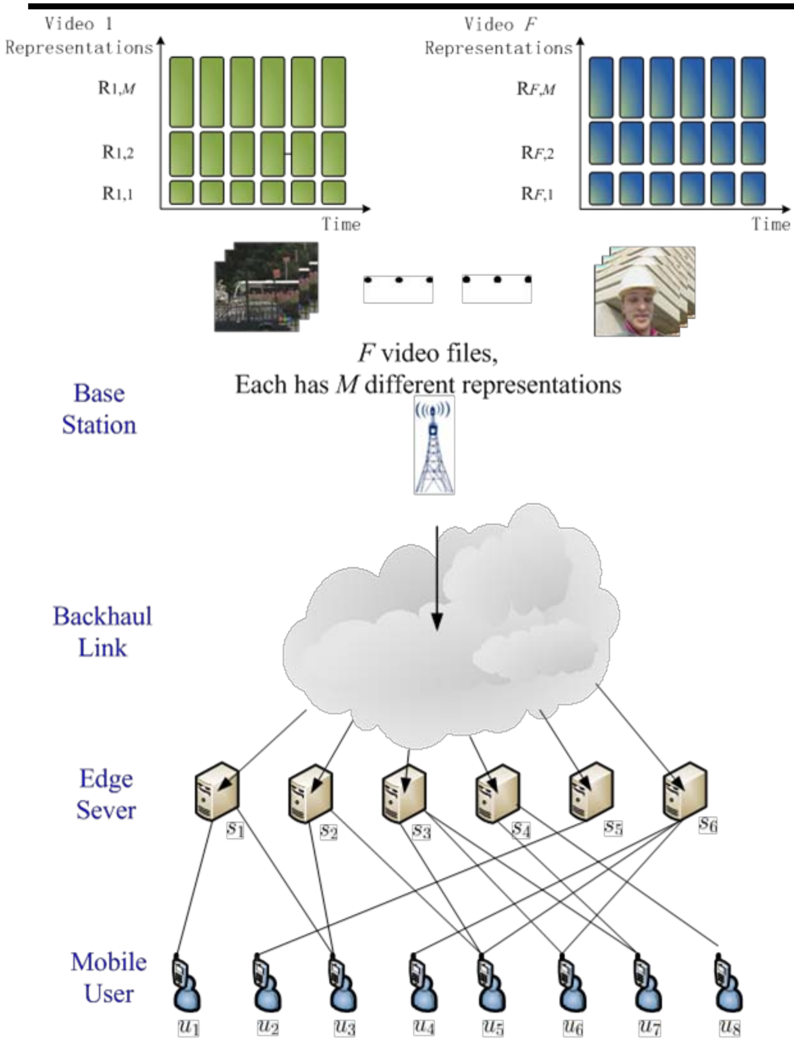
[1] N. Golrezaei, K. Shanmugam, A. G. Dimakis, A. F. Molisch, and G. Caire, "Femtocaching: Wireless video content delivery through distributed caching helpers," in *Proc. IEEE INFOCOM*, 2012, pp. 1107–1115.

[2] N. Golrezaei, K. Shanmugam, A. G. Dimakis, A. F. Molisch, and G. Caire, "Femtocaching: Wireless video content delivery through distributed caching helpers," *IEEE Transactions on Information Theory*, vol. 59, no.12, pp. 8402-8413, Dec. 2013.

Basic assumption (Limitations):

- Each video file has the same size
- No R-D consideration
- No multiple representations for each video file

Practical DASH cache problem

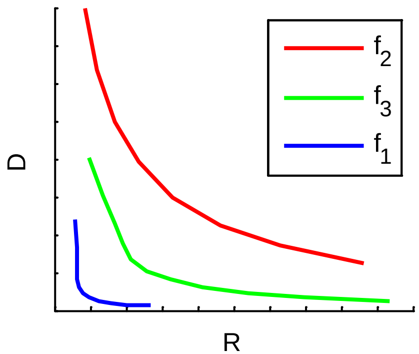


Contributions:

- DASH streaming
  → each video file has multiple representations with different bitrate (file sizes)
  - not only concerned about which video file should be cached at which edge sever
  - also want to know which representation should be selected to cache

- R-D model for different video content
  → for the same bitrate (file size) of different video contents, the delay and distortion are different

Why R-D behavior is important?



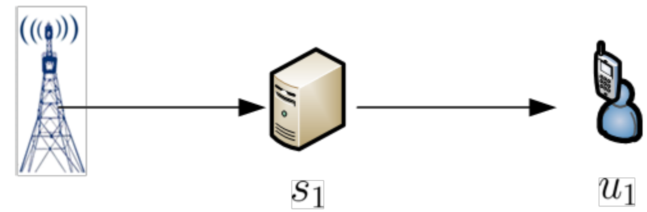
$$f_1: \; P_{f_1} = 0.5, \; R_{f_1} = 150kbps$$

$$f_2: \; P_{f_2} = 0.3, \; R_{f_2} = 400kbps$$

$$f_3: \; P_{f_3} = 0.2, \; R_{f_3} = 300kbps$$

Cache Capacity

$B_s = 500kbps \times \text{duration}$



Motivating Example:

- The simplest one cache to one user topology
- 3 videos, each with one representation and the same video quality (same distortion)
- Popularity based caching strategy without R-D consideration:
  - Only *f1* cached → 50% of user requirement
- A better strategy:
  - Cache *f1* and *f3* → 70% of user requirement

Why DASH benefits?

Motivating Example:

- Best Cache Strategy: *f1* and *f2*

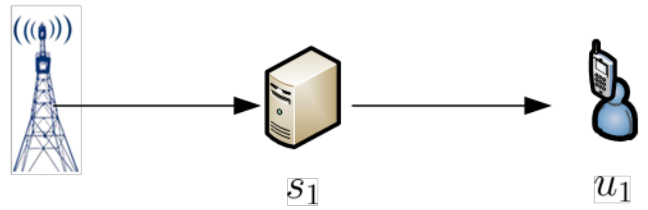
$f_1: \; P_{f_1} = 0.5, \; R_{f_1} = 150kbps$ √

$f_2: \; P_{f_2} = 0.3, \; R_{f_2} = 400kbps$

$f_3: \; P_{f_3} = 0.2, \; R_{f_3} = 300kbps$ √

Cache Capacity

$B_s = 500kbps \times \text{duration}$



DASH Streaming (2 representations):

- Best Cache Strategy: *f1,1* , *f2,2* and *f3,2*

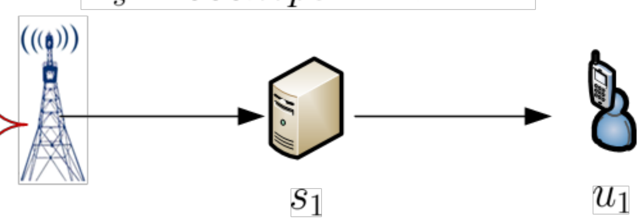
$f_1: \; P_{f_1} = 0.5, \; R_{f_1,1} = 150kbps$√$, \; R_{f_1,2} = 125kbps$

$f_2: \; P_{f_2} = 0.3, \; R_{f_2,1} = 400kbps, \; R_{f_2,2} = 200kbps$ √

$f_3: \; P_{f_3} = 0.2, \; R_{f_3,1} = 300kbps, \; R_{f_3,2} = 150kbps$ √

Cache Capacity

$B_s = 500kbps \times \text{duration}$



Uncoded caching placement - Criterion of the user reception

- $f_{\mathcal{M}} = \{f_1, f_2, \ldots, f_M\}$: the set of M representations of video f in a decreasing order of encoding bitrate;

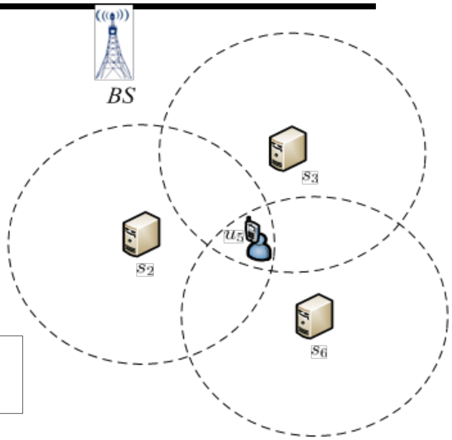
- $\mathcal{S}(u)$: user u's neighborhood of edge servers, sorted in a decreasing order of download link capacity;

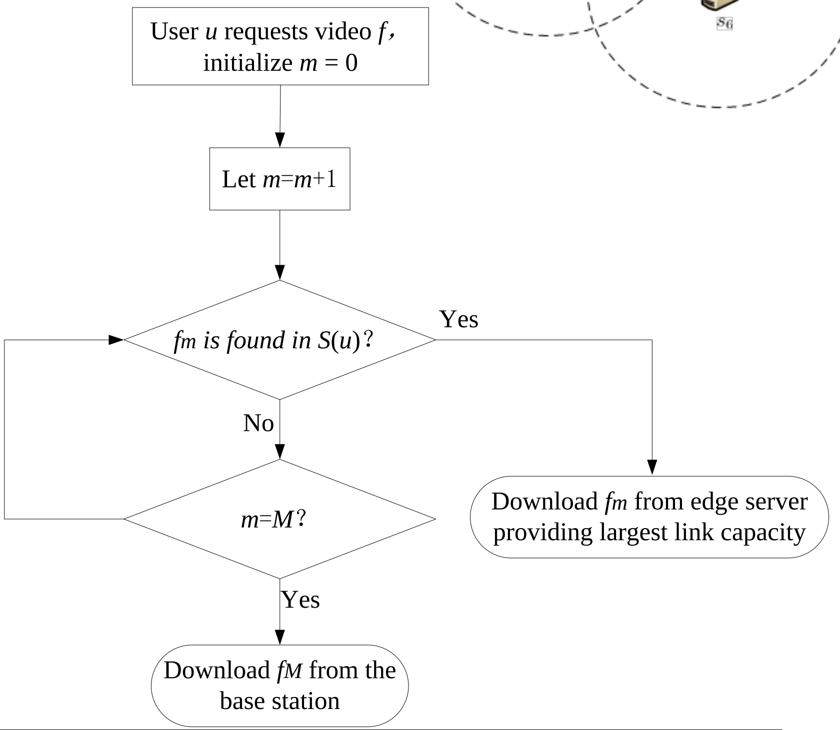
- Indicator variable:

$$a_{f_m,s} = \begin{cases} 1, & \text{if edge server } s \text{ has representation } f_m; \\ 0, & \text{otherwise.} \end{cases}$$

- Distortion reduction:

$$D_{u,f} = \sum_{m=1}^{M} \sum_{i=1}^{|\mathcal{S}(u)|} \left[\prod_{n=1}^{m-1} \prod_{j=1}^{|\mathcal{S}(u)|} (1 - a_{f_n,(j)_u}) \right] \cdot \left[\prod_{j=1}^{i-1} (1 - a_{f_m,(j)_u}) \right] \cdot a_{f_m,(i)_u} \cdot D_f(R_{f_m})$$





Uncoded caching placement - Optimization problem formulation

- F: number of video files; $P_{u,f}$: video request probability; U: number of users;
- S: number of edge severs; B_s: cache capacity of sedge sever s;
- M: number of representations for each video;
- R_{fm}: bitrate for representation m of video file f;
- T: time duration of a video representation;
- Decision variable: $\boldsymbol{A}_{FM\times S} \in \{0,1\}^{FM\times S}$

$$\textbf{P1:} \quad \max_{\boldsymbol{A}_{FM\times S}\in\{0,1\}^{FM\times S}} \sum_{u=1}^{U} \bar{D}_u$$

$$\text{s.t.} \quad 1) \sum_{f=1}^{F}\sum_{m=1}^{M} a_{f_m,s}\cdot R_{f_m}\cdot T \leq B_s, \forall s \in \mathcal{S}$$

- ILP problem, NP hard
- Could be solved in polynomial time if converted to submodular maximization

Average distortion reduction:

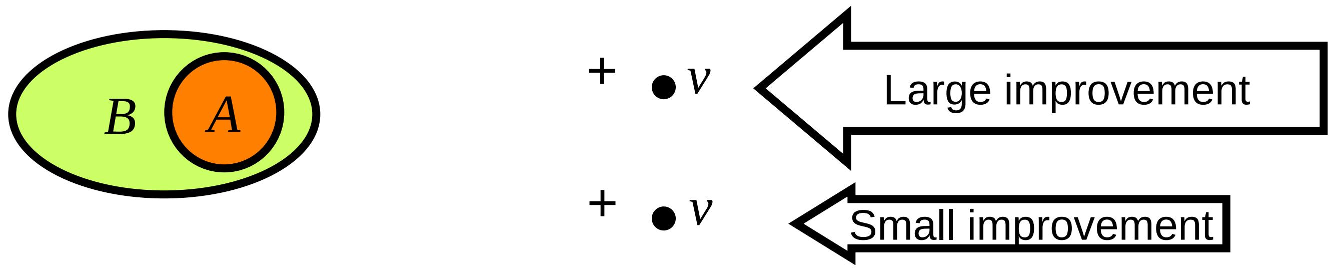
$$\bar{D}_u = \sum_{f=1}^{F}\sum_{m=1}^{M}\sum_{i=1}^{|\mathcal{S}(u)|}\left[\prod_{n=1}^{m-1}\prod_{j=1}^{|\mathcal{S}(u)|}(1-a_{f_n,(j)_u})\right]\cdot\left[\prod_{j=1}^{i-1}(1-a_{f_m,(j)_u})\right]\cdot a_{f_m,(i)_u}\cdot P_{u,f}\cdot D_f(R_{f_m})$$

Submodularity

- Finite ground set $V = \{1,2,\ldots,n\}$
- Set function $G: 2^V \to R$ is submodular iff for any sets $A \subseteq B$, $v \notin B$

$$G(A \cup \{v\}) - G(A) \geq G(B \cup \{v\}) - G(B)$$

Diminishing return:



- Submodular minimization admits polynomial time algorithms
  - Lovász extension, reduced to convex minimization

- Submodular maximization → NP hard
  - Constant factor approximation algorithm

$$G(A_{\text{greedy}}) \geq (1 - 1/e) \max_{|A| \leq k} G(A)$$

DASH Caching- Submodular function maximization

- Define the finite ground set:

$$\mathcal{V} = \{\mathcal{V}_1, \ldots, \mathcal{V}_s, \ldots \mathcal{V}_S\}$$
$$\mathcal{V}_s = \{v^s_{1,1}, \ldots, v^s_{1,M}, \ldots, v^s_{f,m}, \ldots, v^s_{F,1}, \ldots, v^s_{F,M}\}, \ \forall s \in \mathcal{S}$$

- The equivalent objective set function:

$$\max_{\mathcal{A} \subseteq \mathcal{V}} D(\mathcal{A}) = \sum_{u=1}^{U} \bar{D}_u(\mathcal{A})$$

Monotone submodular set function

$$\bar{D}_u(\mathcal{A}) = \sum_{f=1}^{F} \sum_{m=1}^{M} \sum_{i=1}^{|\mathcal{S}(u)|} \left[\prod_{n=1}^{m-1} \prod_{j=1}^{|\mathcal{S}(u)|} \left(1 - \mathbf{1}|_{v^{(j)u}_{f,n} \in \mathcal{A}}\right) \right] \cdot \left[\prod_{j=1}^{i-1} \left(1 - \mathbf{1}|_{v^{(j)u}_{f,m} \in \mathcal{A}}\right) \right] \cdot \mathbf{1}|_{v^{(i)u}_{f,m} \in \mathcal{A}} \cdot P_{u,f} \cdot D_f(R_{f_m})$$

- The cache storage capacity constraint:

$$\text{subject to: } \mathcal{A} \in \mathcal{I}, \ \text{where } \mathcal{I} = \left\{ \mathcal{A}' \subseteq \mathcal{V} \middle| \sum_{f=1}^{F} \sum_{m=1}^{M} \mathbf{1}|_{v^s_{f,m} \in \mathcal{A}'} \cdot R_{f_m} \cdot T \leq B_s, \forall s \in \mathcal{S} \right\}$$

Knapsack constraints

Cost-benefit greedy algorithm

- Maximizing a submodular set function subject to *S* knapsack constraints, each of which takes effect on a subset of the ground set

$$\max_{\mathcal{A}\subseteq\mathcal{V}}\left\{D(\mathcal{A}):\sum_{v^s_{f,m}\in\mathcal{A}}\cdot R_{f_m}\cdot T\leq B_s,\forall s\in\mathcal{S}\right\}$$

- It is proved by [3] that for the special case of one knapsack constraint over the finite ground set, the cost-benefit greedy algorithm that enumerates all initial sets with cardinality 3 can achieve *1-1/e* approximation of the optimal solution.

$$\mathcal{A}^{t+1}=\mathcal{A}^t\cup\left\{\arg\max_{v^{s_t}_{f_t,m_t}\in\mathcal{V}\setminus\mathcal{A}^t:R_{f_t,m_t}\cdot T\leq B-\sum_{v^s_{f,m}\in\mathcal{A}^t}\cdot R_{fm}\cdot T}\frac{D(\mathcal{A}^t\cup\{v^{s_t}_{f_t,m_t}\})-D(\mathcal{A}^t)}{R_{f_t,m_t}T}\right\}$$

[3] M. Sviridenko, "A note on maximizing a submodular set function subject to a knapsack constraint," *Operations Research Letters*, vol. 32, no. 1, pp. 41–43, 2004.

Cost-benefit greedy algorithm – cont'd

- Maximizing a submodular set function subject to a set of S knapsack constraints:

$$\max_{\mathcal{A}\subseteq\mathcal{V}}\left\{D(\mathcal{A}) : \sum_{v^s_{f,m}\in\mathcal{A}} \cdot R_{f_m}\cdot T \le B_s, \forall s \in \mathcal{S}\right\}$$

- k: the cardinality of the initial set
  - k increases
  - approximation performance improves
  - running time also increases

Algorithm 1 k-Cost benefit (k-CB) greedy algorithm

For all initial sets $\mathcal{A}^0 \subseteq \mathcal{V}$ such that $|\mathcal{A}^0| = k$, implement the following cost benefit greedy procedure:

Initialization:

1) Set $\mathcal{V}^0 = \mathcal{V}$ and $t = 1$.

Greedy Search Iteration: (at step $t = 1, 2, 3, \dots$)

1) Given a partial solution $\mathcal{A}^{t-1}$, find

$$\boxed{\theta_t = \max_{v^s_{f,m}\in\mathcal{V}^{t-1}\setminus\mathcal{A}^{t-1}} \frac{D(\mathcal{A}^{t-1}\cup\{v^s_{f,m}\}) - D(\mathcal{A}^{t-1})}{R_{f_m}\cdot T} \qquad (5)}$$

with

$$\boxed{v^{s_t}_{f_t,m_t} = \arg\max_{v^s_{f,m}\in\mathcal{V}^{t-1}\setminus\mathcal{A}^{t-1}} \frac{D(\mathcal{A}^{t-1}\cup\{v^s_{f,m}\}) - D(\mathcal{A}^{t-1})}{R_{f_m}\cdot T} \qquad (6)}$$

Update and Determination:

1) Set $\mathcal{A}^t = \mathcal{A}^{t-1}\cup\{v^{s_t}_{f_t,m_t}\}$, and $\mathcal{V}^t = \mathcal{V}^{t-1}$, if

$$\sum_{f=1}^{F}\sum_{m=1}^{M} 1|_{v^{s_t}_{f,m}\in(\mathcal{A}^{t-1}\cap\mathcal{V}_{s_t})\cup\{v^{s_t}_{f_t,m_t}\}}\cdot R_{f_m}\cdot T \le B_{s_t}; \qquad (7)$$

otherwise, set $\mathcal{A}^t = \mathcal{A}^{t-1}$, and $\mathcal{V}^t = \mathcal{V}^{t-1}\setminus\{v^{s_t}_{f_t,m_t}\}$.

2) If $\mathcal{V}^t\setminus\mathcal{A}^t \neq \emptyset$, set $t = t+1$ and return to the greedy search iteration; otherwise, stop the iteration.

The solution is obtained and output as $\mathcal{A}$, which has the largest value of the objective function $D(\mathcal{A}) = \sum_{u=1}^{U} D_u(\mathcal{A})$ over all the possible choices of the initial sets $\mathcal{A}^0 \subseteq \mathcal{V}$.

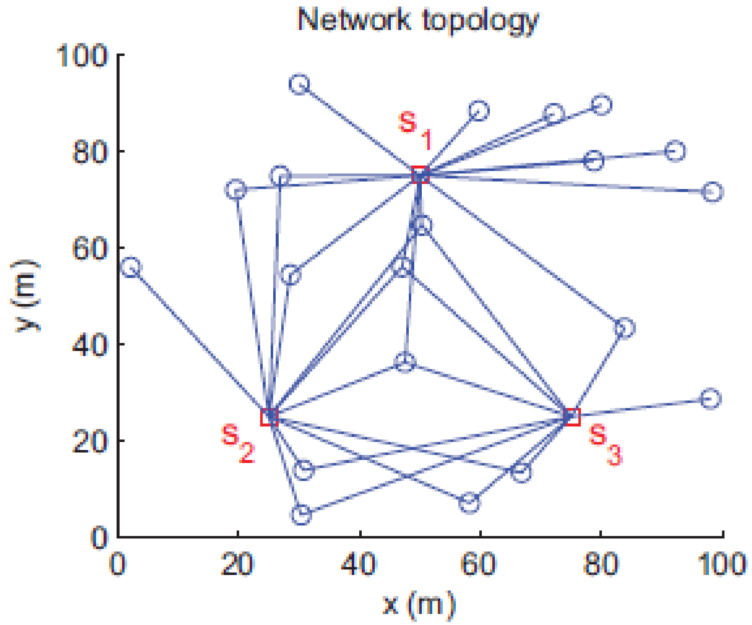
Experiment settings

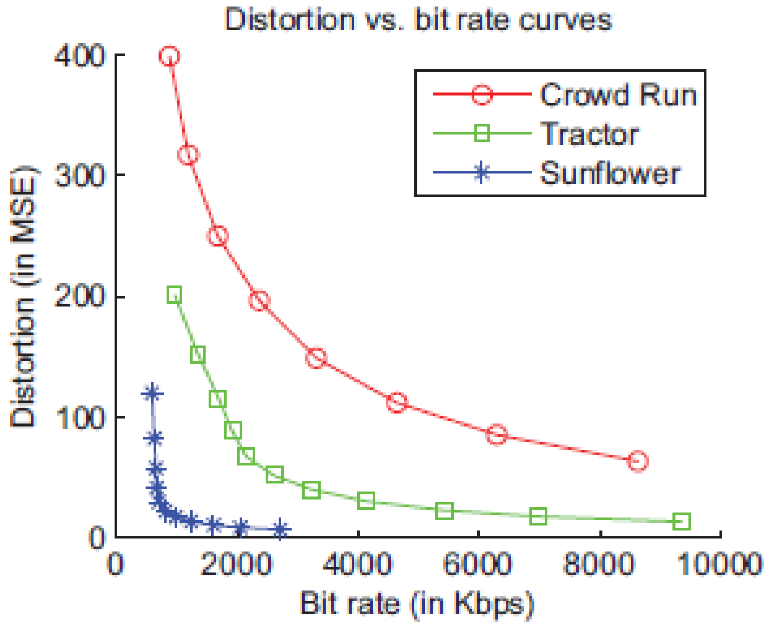
- An illustrative network:

  S=3 edge servers, U=20 mobile users

- Video sources:

  F=3 videos, each has M=3 representations, Zipf distribution





Performance comparisons

Table 2: Comparison on computational complexity and performance of different algorithms

Algorithm	Running time (s)	Computational complexity	$\frac{1}{U}\sum_{u=1}^{U}\bar{D}_u$
Exhaustive search	2068.9	Exponential	361.8
3-CB Greedy	301.2	$O((SFM)^5U)$	361.8
2-CB Greedy	38.2	$O((SFM)^4U)$	361.8
1-CB Greedy	2.6	$O((SFM)^3U)$	357.4
0-CB Greedy	0.3	$O((SFM)^2U)$	347.5
Femto-Greedy	0.3	$O((SFM)^2U)$	312.0
Popular-Cache	0.05	$O(SFM)$	270.6

0.99 – approx.
0.96 – approx.
0.86 – approx.
0.75 – approx.

Placement strategy

Table 1: Distortion reduction (in MSE) after decoding representations of different video sequences

Bit rate	3000 Kbps	2000 Kbps	1000 Kbps
Crowd Run	335.9	275.4	133.3
Tractor	456.3	419.8	303.7
Sunflower	494.6	491.9	483.2

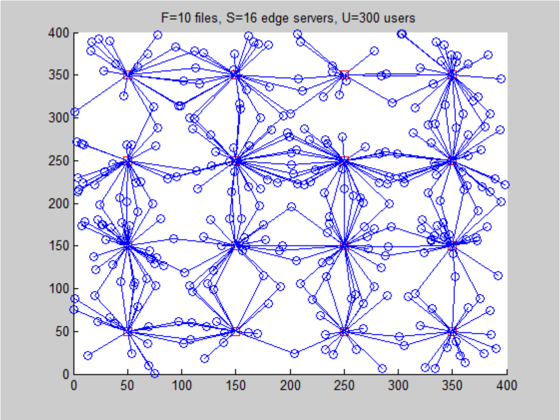
Dependent on video contents

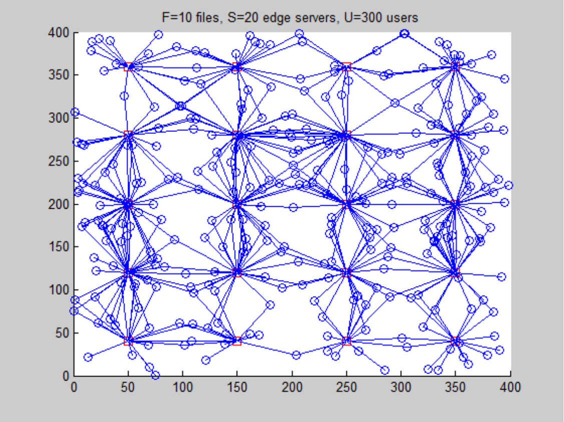
Table 3: Placement strategy for edge servers $S_1 - S_3$ obtained by different algorithms

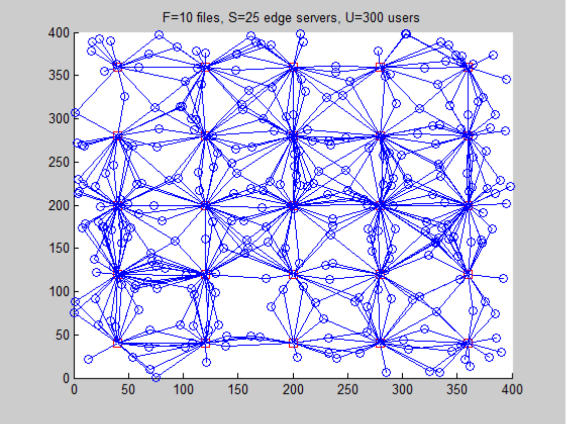
Algorithm	S_1	S_2	S_3
Optimum	$\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$
Femto.	$\begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$
Popular.	$\begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$

Experiments on larger settings

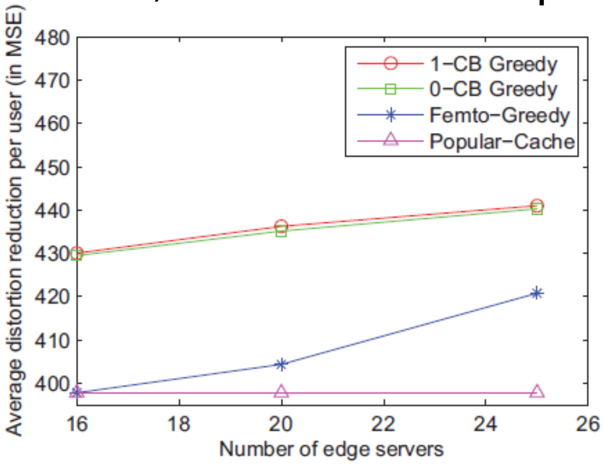
- Network topology: S=16/20/25 edge servers, U=300 mobile users







- Video sources: F=10 videos, each has M=3 representations



Conclusions

- A distributed wireless video caching placement problem for dynamic adaptive streaming
  - Based on content information, R-D perspective
- Submodular maximization with approximation algorithm
  - Polynomial time complexity, theoretical approximation guarantee
- Future work:
  - Take into account more QoE metrics (e.g., startup delay, video quality variation)
  - Coded caching placement and delivery strategy

Thanks!

Q & A