CABaRet: Leveraging Recommendation Systems for Mobile Edge Caching

Savvas Kastanakis
Pavlos Sermpezis
Vasileios Kotronis
Xenofontas Dimitropoulos

FORTH & University of Crete
Greece
Mobile edge caching

Win-Win (user & network): reduces access latency & network load

Low cache hit ratio (CHR)
- small caches (size ~GB vs. catalog size ~PB)
- caching algorithms limitations (variable traffic, frequent changes of users)
A solution: Leverage recommendation systems

- **Why recommendation systems (RS)?**
  - Integrated in *popular* services (YouTube, Netflix, Spotify, etc.)
  - Drive content consumption (~80% in Netflix, >50% in YouTube)

- **How to leverage RS?**
  - Recommend contents that are cached e.g., [ToMM’15, WoWMoM’18]
  - Cache contents that can be recommended e.g., [Globecom’17, JSAC’18]
  - Jointly decide caching and recommendations e.g., [INFOCOM’16]
Initial Recommendations:
- Blue content
- Yellow content

Biased Recommendations:
- Green content
- Yellow content
Caching & Recommendation: An example

Initial Recommendations

Biased Recommendations

P. Sermpezis, MECOMM 2018, “CABaRet: Leveraging Recommendation Systems for Mobile Edge Caching”
Caching & Recommendation: An example

Initial Recommendations

Biased Recommendations
Limitations (or, challenges) & Contributions

- Joint caching and recommendation, needs control / information about:
  - cached contents (i.e., caching)
  - content relations / user preferences (i.e., “good” recommendations)

- **Who controls recommendations?** → content provider
- **Who controls caching?** → network operator or content provider (e.g. MVNO)
- **Who cares about network load?** → network operator

Existing approaches for joint caching and recommendation, require collaboration between network operator & content provider

Our approach / contributions

- only network operator, **without collaboration** with content provider
- practical system & recommendations  
  *(i.e., we did a prototype, it works!)*
- performance evaluation with experiments  
  *(i.e., it works well!)*
- Lightweight system (e.g., mobile app)
- Run only by the network operator (or, even the user)
- Here we focus on YouTube, but it can be generic (for Netflix, Spotify, etc.)
System overview: User-Interface

- **User-Interface (UI)**
  - search bar
  - video player
  - recommendations list
  - etc.
● **Back-end**
  ○ retrieve list of cached video IDs (e.g., from network operator or content provider)
  ○ stream videos to UI
Recommendation Module

- retrieve publicly available information → i.e., no collaboration (from the content provider’s recommendation system, e.g., YouTube API)
- retrieve the list of cached contents (from the back-end)
- build a new recommendation list of related & cached contents
The recommendation algorithm (CABaRet)

1. a user watches a video $v$
2. retrieve from the YouTube API the list of videos related to $v$; let this list be $L$
3. for each videos in $L$, retrieve its related videos, and add them to $L$
4. final list $L$: contains many videos (directly or indirectly) related to $v$
5. retrieve the list of cached videos $C$
6. recommend $N$ videos that are both in $L$ (i.e., related) and $C$ (i.e., cached)
CABaRet characteristics

- Input: video $v$, BFS depth $D$ and width $W$, #recommendations $N$
- Output: list of recommended videos $\sim L \cap C$

- Tuning
  - we want large $L \rightarrow$ more videos, more options for recommendations
    - $|L| = W + W^2 + \ldots + W^D$ (e.g., $W=50$, $D=2 \rightarrow |L|=2550$)
    - larger $W$, $D \rightarrow$ larger $L$
  - we want “good” recommendations
    - larger $D \rightarrow$ videos less related to $v$

- High-quality recommendations
  - $D=1$: directly related/recommended videos
  - $D=2$: indirectly related videos ... e.g., if $a \rightarrow b$ and $b \rightarrow c$, then $a \rightarrow c$

<table>
<thead>
<tr>
<th>$W$</th>
<th>10</th>
<th>20</th>
<th>50</th>
</tr>
</thead>
<tbody>
<tr>
<td>Related videos overlap (at $D=1$ and $D=2$)</td>
<td>70%</td>
<td>85%</td>
<td>92%</td>
</tr>
</tbody>
</table>
Performance evaluation

- Experiments over YouTube service
  - **Caching**: top $C$ most popular contents in a region
  - **Recommendations**: YouTube or CABaRet with $W$ and $D$
  - **User demand**: starts from a popular content, and follows one of the $N$ recommendations; *uniformly* or preference to order of appearance (*Zipf*)

![Graph showing comparison of Cache Hit Ratio (CHR) for different methods and settings.]

- CHR (YouTube) 5-10%
- CHR (CABaRet, $W=50$, $D=2$) 35-90%
  - up to **8-10 times higher** CHR than YouTube
- Even for $D=1$ (high recommendation quality)
  - 2-6 times higher CHR than YouTube
  - ~2 times higher than Reordering [ToMM’15]
CABaRet + Caching optimization

- What if the network operator controls caching as well?
  - Further improvement in CHR
  - How? → optimize caching + then apply CABaRet recommendations

Optimization problem
- for a content \( v \): CABaRet calculates \( L(v) \) and recommends \( \{L(v)\} \cap \{C\} \)
- find \( C \) that maximizes CHR, i.e., \( \sim \{L(v)\} \cap \{C\} \) for all \( v \)

Optimization algorithm
- **NP-hard problem (max set cover)**
- **submodular + monotone**
- greedy algorithm: \((1-1/e)\) approximation
CABaRet + Caching optimization: Results

- Parameters: $N=20$, uniform, $W_{BFS}=20$, $D_{BFS}=2$

- CABaRet: Greedy caching vs. Most popular caching
  - more than 2 times higher CHR

Total gains:
- CABaRet vs. YouTube: 8-10 times higher CHR
- CABaRet + greedy vs. YouTube: $2*(8-10)$ times higher CHR
The problem

- Caching alone is not enough → leverage recommendation systems
- Existing approaches require collaboration of network operator & content provider

The contributions

- Our approach: enable caching & recommendation by the network operator
  - no collaboration with the content provider (only public information)
- Practical recommendation algorithm: CABaRet
- Significant gains in practice (experiments over YouTube)
  - 8-10 times higher CHR due to recommendations
  - extra 2 times higher CHR due to caching

Future work

- Experiments with real users:
  - “Can you tell the difference between YouTube and CABaRet recommendations?... do you like them?”
  - Test it here!!

Pavlos Sermpezis
<sermpezis@ics.forth.gr>