Big Data Meets Telcos: A Proactive Caching Perspective

Ejder Baştug, Mehdi Bennis, Engin Zeydan, Manhal Abdel Kader, Alper Karatepe, Ahmet Salih Er, and Mérouane Debbah

Large Networks and Systems Group (LANEAS), CentraleSupélec, Université Paris-Saclay, Gif-sur-Yvette, France
Centre for Wireless Communications, University of Oulu, Finland
AveaLabs, Istanbul, Turkey
Mathematical and Algorithmic Sciences Lab, Huawei France R&D, Paris, France

ejder.bastug@centralesupelec.fr, bennis@ee.oulu.fi,
{engin.zeydan, alper.karatepe, ahmetsalih.er}@avea.com.tr,
merouane.debbah@huawei.com manhalak@gmail.com

The ETSI Workshop “From Research to Standardization”
May 10 2016, Sophia Antipolis, FRANCE.
Introduction

Motivation

- Mobile cellular networks are becoming increasingly complex [And+14]
- Classical deployment/optimization techniques and solutions (i.e., cell densification, acquiring more spectrum, etc.) are cost-ineffective and thus seen as stopgaps
- This calls for development of novel approaches that leverage recent advances in storage/memory, context-awareness, edge/cloud computing, and falls into framework of big data [BBD14]
- The big data has its notorious 4V: velocity, voracity, volume and variety

Based on these motivations, we focus on

Caching at the edge + enable big data!

In particular, our contributions:

- Collect users’ mobile traffic data from a telecom operator
- Characterize content popularity and size distributions
- Exploit machine learning tools and investigate gains of caching in terms of users’ satisfaction and backhaul offloading


Introduction

Motivation

▶ Mobile cellular networks are becoming increasingly complex [And+14]
▶ Classical deployment/optimization techniques and solutions (i.e., cell densification, acquiring more spectrum, etc.) are cost-ineffective and thus seen as stopgaps
▶ This calls for development of novel approaches that leverage recent advances in storage/memory, context-awareness, edge/cloud computing, and falls into framework of big data [BBD14]
▶ The big data has its notorious 4V: velocity, voracity, volume and variety

Based on these motivations, we focus on

Caching at the edge + enable big data!

In particular, our contributions:

▶ Collect users’ mobile traffic data from a telecom operator
▶ Characterize content popularity and size distributions
▶ Exploit machine learning tools and investigate gains of caching in terms of users’ satisfaction and backhaul offloading


Introduction

Motivation

- Mobile cellular networks are becoming increasingly complex [And+14]
- Classical deployment/optimization techniques and solutions (i.e., cell densification, acquiring more spectrum, etc.) are cost-ineffective and thus seen as stopgaps
- This calls for development of novel approaches that leverage recent advances in storage/memory, context-awareness, edge/cloud computing, and falls into framework of big data [BBD14]
- The big data has its notorious 4V: velocity, voracity, volume and variety

Based on these motivations, we focus on

Caching at the edge + enable big data!

In particular, our contributions:

- Collect users’ mobile traffic data from a telecom operator
- Characterize content popularity and size distributions
- Exploit machine learning tools and investigate gains of caching in terms of users’ satisfaction and backhaul offloading


Introduction

Motivation

- Mobile cellular networks are becoming increasingly complex [And+14]
- Classical deployment/optimization techniques and solutions (i.e., cell densification, acquiring more spectrum, etc.) are cost-ineffective and thus seen as stopgaps
- This calls for development of novel approaches that leverage recent advances in storage/memory, context-awareness, edge/cloud computing, and falls into framework of big data [BBD14]
- The big data has its notorious 4V: velocity, voracity, volume and variety

Based on these motivations, we focus on

Caching at the edge + enable big data!

In particular, our contributions:

- Collect users’ mobile traffic data from a telecom operator
- Characterize content popularity and size distributions
- Exploit machine learning tools and investigate gains of caching in terms of users’ satisfaction and backhaul offloading


Introduction

Motivation

- Mobile cellular networks are becoming increasingly complex [And+14]
- Classical deployment/optimization techniques and solutions (i.e., cell densification, acquiring more spectrum, etc.) are cost-ineffective and thus seen as stopgaps
- This calls for development of novel approaches that leverage recent advances in storage/memory, context-awareness, edge/cloud computing, and falls into framework of big data [BBD14]
- The big data has its notorious 4V: **velocity, voracity, volume** and **variety**

Based on these motivations, we focus on

> Caching at the edge + enable big data!

In particular, our contributions:

- Collect users' mobile traffic data from a telecom operator
- Characterize content popularity and size distributions
- Exploit machine learning tools and investigate gains of caching in terms of users’ satisfaction and backhaul offloading

---


Introduction

Motivation

- Mobile cellular networks are becoming increasingly complex [And+14]
- Classical deployment/optimization techniques and solutions (i.e., cell densification, acquiring more spectrum, etc.) are cost-ineffective and thus seen as stopgaps
- This calls for development of novel approaches that leverage recent advances in storage/memory, context-awareness, edge/cloud computing, and falls into framework of big data [BBD14]
- The big data has its notorious 4V: velocity, voracity, volume and variety

Based on these motivations, we focus on

Caching at the edge + enable big data!

In particular, our contributions:

- Collect users’ mobile traffic data from a telecom operator
- Characterize content popularity and size distributions
- Exploit machine learning tools and investigate gains of caching in terms of users’ satisfaction and backhaul offloading


Introduction

Some Relevant Works

- Scaling laws for caching in content delivery networks [GPT12]
- FemtoCaching architecture [Gol+13]
- LivingOnTheEdge: Edge caching and content popularity learning aspects [BBD14]
- Deployment aspects of cache-enabled single-tier networks [Baş+15]
- Coded caching gains in physical layer [ZE15]
- MDS and regenerating codes [BGL15][Ped+15]

---


Introduction

Some Relevant Works

- Scaling laws for caching in content delivery networks [GPT12]
- FemtoCaching architecture [Gol+13]
- LivingOnTheEdge: Edge caching and content popularity learning aspects [BBD14]
- Deployment aspects of cache-enabled single-tier networks [Baṣ+15]
- Coded caching gains in physical layer [ZE15]
- MDS and regenerating codes [BGL15][Ped+15]

---


Introduction

Some Relevant Works

- Scaling laws for caching in content delivery networks [GPT12]
- FemtoCaching architecture [Gol+13]
- LivingOnTheEdge: Edge caching and content popularity learning aspects [BBD14]
- Deployment aspects of cache-enabled single-tier networks [Baş+15]
- Coded caching gains in physical layer [ZE15]
- MDS and regenerating codes [BGL15][Ped+15]

---


Introduction

Some Relevant Works

▶ Scaling laws for caching in content delivery networks [GPT12]
▶ FemtoCaching architecture [Gol+13]
▶ LivingOnTheEdge: Edge caching and content popularity learning aspects [BBD14]
▶ Deployment aspects of cache-enabled single-tier networks [Baş+15]
▶ Coded caching gains in physical layer [ZE15]
▶ MDS and regenerating codes [BGL15][Ped+15]

---


Introduction

Some Relevant Works

- Scaling laws for caching in content delivery networks [GPT12]
- FemtoCaching architecture [Gol+13]
- LivingOnTheEdge: Edge caching and content popularity learning aspects [BBD14]
- Deployment aspects of cache-enabled single-tier networks [Baş+15]
- Coded caching gains in physical layer [ZE15]
- MDS and regenerating codes [BGL15][Ped+15]
Introduction

Some Relevant Works

- Scaling laws for caching in content delivery networks [GPT12]
- FemtoCaching architecture [Gol+13]
- LivingOnTheEdge: Edge caching and content popularity learning aspects [BBD14]
- Deployment aspects of cache-enabled single-tier networks [Baṣ+15]
- Coded caching gains in physical layer [ZE15]
- MDS and regenerating codes [BGL15][Ped+15]
Network Model

Scenario

- Our scenario...
Network Model

Scenario

- $N$ user terminals (UTs) from the set $\mathcal{N} = \{1, \ldots, N\}$
Network Model

Scenario

- $M$ small base stations (SBSs) from the set $\mathcal{M} = \{1, \ldots, M\}$
Each SBS $m$ has storage capacity of $S_m$
A wireless link with total capacity of $C'_m$ Mbyte/s
A wired backhaul link with capacity $C_m$ Mbyte/s for SBS $m$

Limited backhaul regime, with $C_m < C'_m$
A library of $F$ contents, where each content $f \in F$

SBSs proactively cache contents from the library $F$ during peak-off hours

Each content $f$ has a size of $L(f)$ Mbyte and bitrate requirement of $B(f)$ Mbyte/s
Network Model

Scenario

- In ordered case, content requests follow a Zipf-like distribution $P_F(f), \forall f \in F$ with shape parameter $\alpha$.
- In unordered case, content popularity matrix $P_m(t) \in \mathbb{R}^{N \times F}$. 

▶ In ordered case, content requests follow a Zipf-like distribution $P_F(f), \forall f \in F$ with shape parameter $\alpha$.
▶ In unordered case, content popularity matrix $P_m(t) \in \mathbb{R}^{N \times F}$. 

4/19
Network Model

Performance Metrics

A request $d$ is called **satisfied** if the rate of content delivery is equal or higher than the bitrate of the content in the end of service, such as:

$$\frac{L(f_d)}{\tau'(f_d) - \tau(f_d)} \geq B(f_d)$$  \hspace{1cm} (1)

where

- $f_d$ describes the requested content
- $L(f_d)$ and $B(f_d)$ are the size and bitrate of the content
- $\tau(f_d)$ is the arrival time of the content request and $\tau'(f_d)$ the end time delivery

Users’ average request satisfaction ratio is then defined for the set of all requests, that is:

$$\eta(D) = \frac{1}{|D|} \sum_{d \in D} \mathbb{1}\left\{ \frac{L(f_d)}{\tau'(f_d) - \tau(f_d)} \geq B(f_d) \right\}$$  \hspace{1cm} (2)

where $\mathbb{1}\{\ldots\}$ is the indicator function. Now, denoting $R_d(t)$ Mbyte/s as the instantaneous rate of backhaul for the request $d$ at time $t$, with $R_d(t) \leq C_m$, $\forall m \in \mathcal{M}$, the average backhaul load is then expressed as:

$$\rho(D) = \frac{1}{|D|} \sum_{d \in D} \frac{1}{L(f_d)} \sum_{t=\tau(f_d)}^{\tau'(f_d)} R_d(t)$$  \hspace{1cm} (3)
Network Model
Performance Metrics

A request $d$ is called **satisfied** if the rate of content delivery is equal or higher than the bitrate of the content in the end of service, such as:

$$\frac{L(f_d)}{\tau'(f_d) - \tau(f_d)} \geq B(f_d)$$  \hspace{1cm} (1)

where

- $f_d$ describes the requested content
- $L(f_d)$ and $B(f_d)$ are the size and bitrate of the content
- $\tau(f_d)$ is the arrival time of the content request and $\tau'(f_d)$ the end time delivery

Users’ average request **satisfaction ratio** is then defined for the set of all requests, that is:

$$\eta(D) = \frac{1}{D} \sum_{d \in D} \mathbb{1}\left\{ \frac{L(f_d)}{\tau'(f_d) - \tau(f_d)} \geq B(f_d) \right\}$$  \hspace{1cm} (2)

where $\mathbb{1}\{...\}$ is the indicator function. Now, denoting $R_d(t)$ Mbyte/s as the instantaneous rate of backhaul for the request $d$ at time $t$, with $R_d(t) \leq C_m$, $\forall m \in \mathcal{M}$, the average **backhaul load** is then expressed as:

$$\rho(D) = \frac{1}{D} \sum_{d \in D} \frac{1}{L(f_d)} \sum_{t=\tau(f_d)}^{\tau'(f_d)} R_d(t)$$  \hspace{1cm} (3)
A request $d$ is called **satisfied** if the rate of content delivery is equal or higher than the bitrate of the content in the end of service, such as:

\[
\frac{L(f_d)}{\tau'(f_d) - \tau(f_d)} \geq B(f_d)
\] (1)

where

- $f_d$ describes the requested content
- $L(f_d)$ and $B(f_d)$ are the size and bitrate of the content
- $\tau(f_d)$ is the arrival time of the content request and $\tau'(f_d)$ the end time delivery

Users’ average request **satisfaction ratio** is then defined for the set of all requests, that is:

\[
\eta(D) = \frac{1}{D} \sum_{d \in D} \mathbb{1} \left\{ \frac{L(f_d)}{\tau'(f_d) - \tau(f_d)} \geq B(f_d) \right\}
\] (2)

where $\mathbb{1} \{ \ldots \}$ is the indicator function. Now, denoting $R_d(t)$ Mbyte/s as the instantaneous rate of backhaul for the request $d$ at time $t$, with $R_d(t) \leq C_m$, $\forall m \in M$, the average **backhaul load** is then expressed as:

\[
\rho(D) = \frac{1}{D} \sum_{d \in D} \frac{1}{L(f_d)} \sum_{t=\tau(f_d)}^{\tau'(f_d)} R_d(t)
\] (3)
Network Model

Backhaul Offloading Problem

\[
\begin{align*}
\text{minimize} & \quad \rho(D) \\
\text{subject to} & \quad L_{\text{min}} \leq L(f_d) \leq L_{\text{max}}, \quad \forall d \in D, \\
& \quad B_{\text{min}} \leq B(f_d) \leq B_{\text{max}}, \quad \forall d \in D, \\
& \quad R_d(t) \leq C_m, \quad \forall t, \forall d \in D, \forall m \in M, \\
& \quad R'_d(t) \leq C'_m, \quad \forall t, \forall d \in D, \forall m \in M, \\
& \quad \sum_{f \in F} L(f)x_{m,f}(t) \leq S_m, \quad \forall t, \forall m \in M, \\
& \quad \sum_{n \in N} \sum_{f \in F} P_{n,f}(t) = 1, \quad \forall t, \forall m \in M, \\
& \quad x_{m,f}(t) \in \{0, 1\}, \quad \forall t, \forall f \in F, \forall m \in M, \\
& \quad \eta_{\text{min}} \leq \eta(D),
\end{align*}
\]

where \(R'_d(t)\) Mbyte/s describes the instantaneous rate of wireless link for request \(d\) and \(\eta_{\text{min}}\) represents the minimum target satisfaction ratio.

We simplify (and solve) this problem by first estimating \(P\), then caching contents greedily (represented by \(X\)).
Network Model

Backhaul Offloading Problem

\[
\begin{align*}
\text{minimize} & \quad \rho(D) \quad (4) \\
\text{subject to} & \quad L_{\min} \leq L(f_d) \leq L_{\max}, \quad \forall d \in D, \quad (4a) \\
& \quad B_{\min} \leq B(f_d) \leq B_{\max}, \quad \forall d \in D, \quad (4b) \\
& \quad R_d(t) \leq C_m, \quad \forall t, \forall d \in D, \forall m \in M, \quad (4c) \\
& \quad R'_d(t) \leq C'_m, \quad \forall t, \forall d \in D, \forall m \in M, \quad (4d) \\
& \quad \sum_{f \in F} L(f)x_{m,f}(t) \leq S_m, \quad \forall t, \forall m \in M, \quad (4e) \\
& \quad \sum_{n \in N} \sum_{f \in F} P_{n,f}(t) = 1, \quad \forall t, \forall m \in M, \quad (4f) \\
& \quad x_{m,f}(t) \in \{0, 1\}, \quad \forall t, \forall f \in F, \forall m \in M, \quad (4g) \\
& \quad \eta_{\min} \leq \eta(D), \quad (4h)
\end{align*}
\]

where \( R'_d(t) \) Mbyte/s describes the instantaneous rate of wireless link for request \( d \) and \( \eta_{\min} \) represents the minimum target satisfaction ratio.

We simplify (and solve) this problem by first estimating \( P \), then caching contents greedily (represented by \( X \))
Network Model

Content Popularity Learning

If sufficient amount of users’ ratings are available at the SBSs, we can construct a $k$-rank approximate popularity matrix $P \approx NTF$, by jointly learning the factor matrices $N \in \mathbb{R}^{k \times N}$ and $F \in \mathbb{R}^{k \times F}$ that minimizes the following cost function:

$$\min_{P} \sum_{P_{ij} \in \mathcal{P}} \left( n_i^T f_j - P_{ij} \right)^2 + \mu \left( \|N\|_F^2 + \|F\|_F^2 \right)$$

(5)

where

- The summation is done over the corresponding user/content rating pairs $P_{ij}$ in the training set $\mathcal{P}$
- The vectors $n_i$ and $f_j$ here describe the $i$-th and $j$-th columns of $N$ and $F$ matrices respectively
- $\|.\|_F^2$ represents the Frobenius norm
- The parameter $\mu$ is used to provide a balance between the regularization and fitting the training data
Mobile Traffic: Approximately over 80 TByte of total data flowing in uplink/downlink daily in the mobile operator’s core network

The big data platform runs in the operator’s core network, and collects the data from several base stations

Collected Data: Approximately 7 hours starting from 12 pm to 7 pm on Saturday 21’st of March 2015

Big Data Platform: Cloudera’s Distribution Including Apache Hadoop (CDH4) version on four nodes including one cluster name node

Each node node with INTEL Xeon CPU E5-2670 running @2.6 GHz, 32 Core CPU, 132 GByte RAM, 20 TByte hard disk
Mobile Traffic: Approximately over 80 TByte of total data flowing in uplink/downlink daily in the mobile operator’s core network

The big data platform runs in the operator’s core network, and collects the data from several base stations

Collected Data: Approximately 7 hours starting from 12 pm to 7 pm on Saturday 21’st of March 2015

Big Data Platform: Cloudera’s Distribution Including Apache Hadoop (CDH4) version on four nodes including one cluster name node

Each node node with INTEL Xeon CPU E5-2670 running @2.6 GHz, 32 Core CPU, 132 GByte RAM, 20 TByte hard disk
**Mobile Traffic:** Approximately over 80 TByte of total data flowing in uplink/downlink daily in the mobile operator’s core network

**The big data platform runs in the operator’s core network, and collects the data from several base stations**

**Collected Data:** Approximately 7 hours starting from 12 pm to 7 pm on Saturday 21’st of March 2015

**Big Data Platform:** Cloudera’s Distribution Including Apache Hadoop (CDH4) version on four nodes including one cluster name node

**Each node node with INTEL Xeon CPU E5-2670 running @2.6 GHz, 32 Core CPU, 132 GByte RAM, 20 TByte hard disk**
Mobile Traffic: Approximately over 80 TByte of total data flowing in uplink/downlink daily in the mobile operator’s core network.

The big data platform runs in the operator’s core network, and collects the data from several base stations.

Collected Data: Approximately 7 hours starting from 12 pm to 7 pm on Saturday 21’st of March 2015.

Big Data Platform: Cloudera’s Distribution Including Apache Hadoop (CDH4) version on four nodes including one cluster name node.

Each node node with INTEL Xeon CPU E5-2670 running @2.6 GHz, 32 Core CPU, 132 GByte RAM, 20 TByte hard disk.
- **Mobile Traffic**: Approximately over 80 TByte of total data flowing in uplink/downlink daily in the mobile operator’s core network

- The big data platform runs in the operator’s core network, and collects the data from several base stations

- **Collected Data**: Approximately 7 hours starting from 12 pm to 7 pm on Saturday 21’st of March 2015

- **Big Data Platform**: Cloudera’s Distribution Including Apache Hadoop (CDH4) version on four nodes including one cluster name node

  - Each node node with INTEL Xeon CPU E5-2670 running @2.6 GHz, 32 Core CPU, 132 GByte RAM, 20 TByte hard disk
Mobile Traffic: Approximately over 80 TByte of total data flowing in uplink/downlink daily in the mobile operator’s core network

The big data platform runs in the operator’s core network, and collects the data from several base stations

Collected Data: Approximately 7 hours starting from 12 pm to 7 pm on Saturday 21’st of March 2015

Big Data Platform: Cloudera’s Distribution Including Apache Hadoop (CDH4) version on four nodes including one cluster name node

Each node node with INTEL Xeon CPU E5-2670 running @2.6 GHz, 32 Core CPU, 132 GByte RAM, 20 TByte hard disk
Big Data Platform

Data Extraction Process
Big Data Platform

Data Extraction Process
Big Data Platform

Data Extraction Process

1) Collect Raw Data

Control Packets

Data Packets

High-speed data flow

Traffic mirroring

Cluster computing

Database
Big Data Platform

Data Extraction Process

Location/Session Fields
CELL-ID, SAC, LAC, TEID

Content Request Field
HTTP URI

Request Time Field
FRAME TIME

2) Extract Relevant Fields

Control Packets

Data Packets

1) Collect Raw Data

Database

Cluster computing

Traffic mirroring

high-speed data flow
Big Data Platform

Data Extraction Process

1) Collect Raw Data

2) Extract Relevant Fields

   - Control Packets
   - Data Packets

3) Match Fields

   - Location/Session Fields: CELL-ID, SAC, LAC, TEID
   - Content Request Field: HTTP URI
   - Request Time Field: FRAME TIME

traces-table-temp

<table>
<thead>
<tr>
<th>HTTP URI</th>
<th>FRAME TIME</th>
<th>TEID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

HDFS & MapReduce

Cluster computing

Database

Traffic mirroring

high-speed data flow
Big Data Platform

Data Extraction Process

1) Collect Raw Data
2) Extract Relevant Fields
   - Location/Session Fields: CELL-ID, SAC, LAC, TEID
   - Content Request Field: HTTP URI
   - Request Time Field: FRAME TIME
3) Match Fields
   - traces-table-temp:
     - HTTP URI
     - FRAME TIME
     - TEID
4) Calculate Content Sizes
   - traces-table:
     - HTTP URI
     - FRAME TIME
     - TEID
     - SIZE

HDFS & MapReduce

Traffic mirroring

Cluster computing

Database

Size Calculator (HTTPClient API)
Big Data Platform

Data Extraction Process

1) Collect Raw Data
2) Extract Relevant Fields
3) Match Fields
4) Calculate Content Sizes
5) Store Processed Data

Location/Session Fields
- CELL-ID, SAC, LAC, TEID

Content Request Field
- HTTP URI

Request Time Field
- FRAME TIME

Control Packets

Data Packets

traces-table-temp
- HTTP URI
- FRAME TIME
- TEID

traces-table
- HTTP URI
- FRAME TIME
- TEID
- SIZE

HDFS & MapReduce

Size Calculator (HTTPClient API)

Database

Cluster computing

Traffic mirroring

High-speed data flow
The popularity behaviour of contents follows a Zipf law with steepness parameter $\alpha = 1.36$.
Figure 2: Cumulative size distribution.

- Total catalog size of 17.7451 GByte
- The cumulative size up to 41-th most-popular contents has 0.1 GByte of size whereas a dramatical increase appears afterwards
## Numerical Results

**Overview**

### Table 1: List of simulation parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T$</td>
<td>Time slots</td>
<td>6 hours 47 minutes</td>
</tr>
<tr>
<td>$D$</td>
<td>Number of requests</td>
<td>422529</td>
</tr>
<tr>
<td>$F$</td>
<td>Number of contents</td>
<td>16419</td>
</tr>
<tr>
<td>$M$</td>
<td>Number of small cells</td>
<td>16</td>
</tr>
<tr>
<td>$L_{\text{min}}$</td>
<td>Min. size of a content</td>
<td>1 Byte</td>
</tr>
<tr>
<td>$L_{\text{max}}$</td>
<td>Max. size of a content</td>
<td>6.024 GByte</td>
</tr>
<tr>
<td>$B(f)$</td>
<td>Bitrate of content $f$</td>
<td>4 Mbyte/s</td>
</tr>
<tr>
<td>$\sum_m C_m$</td>
<td>Total backhaul link capacity</td>
<td>3.8 Mbyte/s</td>
</tr>
<tr>
<td>$\sum_m \sum_n C'_m$</td>
<td>Total wireless link capacity</td>
<td>120 Mbyte/s</td>
</tr>
</tbody>
</table>

**Ground Truth:** The content popularity matrix $P$ is constructed from all available information in *traces-table*. The rating density = 6.42%

**Collaborative Filtering:** The problem in (5) is attempted by first choosing 10% of ratings from *traces-table* uniformly at random. Then, these ratings are used in the training stage of the algorithm and missing entries/ratings of $P$ are estimated via regularized singular-value decomposition (SVD)
Numerical Results

Overview

Table 1: List of simulation parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T$</td>
<td>Time slots</td>
<td>6 hours 47 minutes</td>
</tr>
<tr>
<td>$D$</td>
<td>Number of requests</td>
<td>422529</td>
</tr>
<tr>
<td>$F$</td>
<td>Number of contents</td>
<td>16419</td>
</tr>
<tr>
<td>$M$</td>
<td>Number of small cells</td>
<td>16</td>
</tr>
<tr>
<td>$L_{\text{min}}$</td>
<td>Min. size of a content</td>
<td>1 Byte</td>
</tr>
<tr>
<td>$L_{\text{max}}$</td>
<td>Max. size of a content</td>
<td>6.024 GByte</td>
</tr>
<tr>
<td>$B(f)$</td>
<td>Bitrate of content f</td>
<td>4 Mbyte/s</td>
</tr>
<tr>
<td>$\sum_m C_m$</td>
<td>Total backhaul link capacity</td>
<td>3.8 Mbyte/s</td>
</tr>
<tr>
<td>$\sum_m \sum_n C'_m$</td>
<td>Total wireless link capacity</td>
<td>120 Mbyte/s</td>
</tr>
</tbody>
</table>

- **Ground Truth**: The content popularity matrix $P$ is constructed from all available information in traces-table. The rating density $= 6.42\%$

- **Collaborative Filtering**: The problem in (5) is attempted by first choosing 10% of ratings from traces-table uniformly at random. Then, these ratings are used in the training stage of the algorithm and missing entries/ratings of $P$ are estimated via regularized singular-value decomposition (SVD)
Numerical Results

Users’ Satisfaction

Figure 3: Evolution of satisfaction with respect to the storage size.

- With 40% of storage size, the ground truth achieves 92% of satisfaction whereas the collaborative filtering (CF) has value of 69%.
- There is a performance gap between the ground truth and CF until 87% of storage size, which is due to the estimation errors.
Figure 4: Evolution of backhaul usage with respect to the storage size.

- With 87% of storage size for caching, both approaches offload 98% of backhaul usage.
- Popularity-based storage fails to capture these content size aspects on the backhaul usage.
Figure 5: Evolution of root mean square error (RMSE) with respect to the training density.

- Increasing training density in this setup improves the estimation.
Conclusions

- We have presented a proactive caching approach for 5G wireless networks by exploiting large amount of available data on a big data platform and employing machine learning tools.
- First study on exploitation of big data for caching in wireless networks.
- Performance gains depend on storage size and rating density.

Interesting future directions of this experimental work are:

- Detailed characterization of the traffic.
- Novel machine learning algorithms.
- Deterministic/randomized cache decision algorithms which are not purely based on content popularity and storing most popular contents.
Conclusions

- We have presented a proactive caching approach for 5G wireless networks by exploiting large amount of available data on a big data platform and employing machine learning tools.
- First study on exploitation of big data for caching in wireless networks.
- Performance gains depend on storage size and rating density.

Interesting future directions of this experimental work are:

- Detailed characterization of the traffic.
- Novel machine learning algorithms.
- Deterministic/randomized cache decision algorithms which are not purely based on content popularity and storing most popular contents.
Thanks for your attention

http://www.laneas.com/ejder-bastug
ejder.bastug@centralesupelec.fr

Related Publications


This research has been supported by the ERC Starting Grant 305123 MORE (Advanced Mathematical Tools for Complex Network Engineering), the SHARING project under the Finland grant 128010 and TUBITAK TEYDEB 1509 project grant (numbered 9120067) and the project BESTCOM.


