Paper survey related with web/app performance optimization and MEC

Youngseok Lee
lee@cnu.ac.kr
cnu.lee@ucdavis.edu


Revisiting MEC

- MEC architecture is a new revenue stream for mobile operators that has not matured sufficiently
- A few application areas adopting edge computing
  - Fog computing, AR, content delivery
Research areas

• Computation offloading
  • Edge accelerated Web browsing (EAB) prototype designed for Web application execution using a better offloading technique.
  • Network architecture-based solutions, such as cloud of co-located mobile devices

• Storage
• Low latency
• Energy efficiency
Research on Infrastructure

• Deployment scenarios
  • MECs in outdoor: RAN
  • MECs in indoor: Wifi or 3G/4G access points

• MEC testbeds
  • 5G test network at Oulu, Finland
  • Industrial testbeds: Nokia and China
Other Open Issues

• Security
  • The application data movement: possible with encryption
• Pricing

• Web interface
  • Not optimized for mobile
• Other
  • Privacy, openness, multiservices, robustness, resilience
Content Delivery and Caching

• The edge computing technology plays a key role in website performance optimization
  • caching HTML content
  • reorganizing Web layout
  • resizing Web components
Improving Short-lived Web Traffic Performance

• How to compensate the throughput gap caused by the computation latency during short-lived application loading by only adapting the transport-layer protocol?
  • not affected by any application layer constraints such as HTTPS content encryption and security policy
  • embed network intelligence at mobile edge

• Through the optimization of TCP initial window (IW) size
  • short-lived applications such as webpage downloading, where content downloading is normally completed during the TCP slow-start phase

• Mobile edge’s awareness of the computation time on the device
  • does play an important role in webpage downloading performance
MEC Support

- Optimize TCP IW
- DNS response with context information

Figure 1: Overview of proposed MEC framework

Figure 2: Throughput gap due to computation activity
How MEC Server Obtains Context Info

• Not every web content but popular web pages
• Testing as a service (TaaS)
  • MEC server performs test/measurement services
  • Google firebase or Flywheel
  • Computation latency and total size of content can be obtained
Performance Evaluation

- real LTE-A testing infrastructure
- use QUIC as the underlying protocol
- when the computation latency accounts for less than 20% of the overall downloading time,
  - the throughput gap can be fully compensated.

*Figure 3: Proof-of-concept implementation in LTE-A test bed*
When the proportion of computation latency varies between 20% and 50%,

- the throughput can be improved up to 34.5%. Such improved downloading throughput has led to the reduced webpage downloading time by up to 25.1%.

![Optimal IW (Content Size: 1000KB, α 1/3)](image)

5: Optimal IW in real LTE-A network (varying content size)
Recent Changes in Web

- HTTPS
  - TLS + HTTP
- QUIC
  - UDP + TLS + TCP Congestion Window algorithm
- HTTP/2: low latency protocol
  - Single TCP connection, Server push, Header compression, binary, multiple streams

- TCP initial window
  - IETF RFC 6928 – a proposal to increase the TCP Initial Window to 10 segments: 10 * 1432 bytes = 14KB for the initial web page
Latency vs Bandwidth impact on Page Load Time

Page Load Time as bandwidth increases

Page Load Time as latency decreases

Single digit % perf improvement after 5 Mbps
Linear improvement in page load time!

“To speed up the Internet at large, we should look for more ways to bring down RTT. What if we could reduce cross-atlantic RTTs from 150 ms to 100 ms? This would have a larger effect on the speed of the internet than increasing a user’s bandwidth from 3.9 Mbps to 10 Mbps or even 1 Gbps.” - Mike Belshe

bit.ly/http2-opt
Optimizing Web

Cache resources on the client
Redundant data transfers are... redundant!
- Cache-Control and ETag’s on each resource is a must.

Compress assets during transfer
Bytes are slow and expensive to transfer...
- GZIP offers 40-80% savings on most assets - easy win.

If/when lower layers fail, we’re forced to “optimize” at the application layer...
Handshakes, goodput, packet loss, ...

Reuse TCP connections
Connection are expensive
- handshake latency, resource overhead, ...

Resource fetch, execution and processing, ...

Use a Content Delivery Network
Page rendering is latency-bound (most of the time)
- lower roundtrip times are critical to optimize asset delivery

Parallelism, prioritization, protocol overhead, ...
All things DNS (and QUIC :))

Reduce DNS lookups
Unresolved names block requests

Link layer
(Ethernet, WiFi, LTE...)

Application

HTTP

TCP
UDP

RRC and radio delays, energy consumption, ...

bit.ly/http2-opt
Questions about supporting mobile web/app performance by MEC server

- TCP configuration?
  - Scalability issue for every web domain and TCP connections
  - Security issue for kernel-level TCP configuration
    - Root privilege is required
- QUIC configuration
  - Application layer configuration: CUBIC + TLS + UDP
  - Only by Google servers
Discussion

• Improving end-to-end application performance by middle box (MEC/proxy/CDN)
  • Challenges
    • Encryption: TLS, (DNS)
    • Security: certificate pinning, HSTS
    • Scalability
  • Possibilities
    • Caching/proxy: Amazon Silk, Opera mini
    • CDN: Akamai/Limelight, Netflix
    • HTTP/2 optimization with TaaS
      • DNS, server push, compression, concatenating resources, inline resources

Speeding up Web Page Loads with Shandian, USENIX 2016