E2E: Embracing User Heterogeneity to Improve Quality of Experience on the Web

Xu Zhang¹, Siddhartha Sen², Daniar Kurniawan¹, Haryadi Gunawi¹, Junchen Jiang¹

¹University of Chicago, ²Microsoft Research
Page load time matters!

How One Second Could Cost Amazon $1.6 Billion In Sales
https://www.fastcompany.com/.../how-one-second-could-cost-amazon-16-billion-sales

Users are happier with faster page load time!

How Website Performance Affects Conversion Rates | Cloudflare
https://www.cloudflare.com/learning/.../more/website-performance-conversion-rates/

Impact on your entire marketing and sales pipelines. Up to 79% of customers who are dissatisfied with a website's performance say they’re less likely to buy from the same site again. But can big business issues actually be solved simply
Conventional wisdom

Cut all server-side processing delays
- Minimize mean delay
- Minimize P99 delay
- Minimize rate of missing a deadline
Q: Should we treat all requests in the SAME way?
Experiment: Observe the difference in the quality of page load events
Can you see significant improvement?
Set #A
Set #A: Before improvement
Set #A: After improvement
Can you see significant improvement?
Set #B
Set #2: Before improvement
Set #2: After improvement
Can you see significant improvement?
Does it mean Set #B has a bigger delay reduction?
No! The two sets improved by the \textit{SAME} amount of delay!
Same delay reduction, but different improvements

In sensitive region, people are more sensitive to additional delay.
Requests have different sensitivities to additional delay

- too fast to matter
- sensitive
- too slow to matter
Requests have different sensitivities to additional delay

Analysis from Microsoft online store traces and user study on MTurk

- too fast to matter
- sensitive
- too slow to matter
Idea: Focusing on more sensitive requests

Conventional

Treating requests equally

E2E

Focusing on more sensitive requests
Data center without E2E

Request (browser) → WAN (last-mile, ISP) → Data center → Shared-resource service

- External delay
- Server-side delay
- Total delay

August 22, 2019
Data center with E2E

External delay

Resource allocation decision

Server-side delay

Total delay
Potential gain

- We reshuffle the server-side delays between concurrent requests
- More sensitive requests get smaller delays

![Graph showing 20% higher QoE and 40% higher throughput with E2E compared to Default]
Our opportunity

- Current content providers do not distinguish the requests.
Case study: replica selection

- Assign sensitive requests to the fast replica

Default policy: Load balanced
Case study: replica selection

• Assign sensitive requests to the fast replica

Default policy: Load balanced

Sensitive requests
Insensitive requests

E2E: Unbalanced load distribution

August 22, 2019
How do we decide a request’s sensitivity?

- **Goal**: Sensitive requests will be sent to fast replicas
- **Challenge**: a request’s sensitivity is not an inherent property
  - Strawman: A request’s sensitivity is the slope of this request’s external delay
- **Observation**: The optimal replica selection depends on the server-side delay distribution.

Server-side delay distribution

August 22, 2019
How to select replicas for heterogenous requests?

- Send requests to replicas
- Maximize $\sum_i QoE(\text{external\_delay}_i + \text{server\_delay}_i)$
- Classical maximum bipartite graph matching problem
Need to reduce the decision-making overhead!

Reduce the time consumption of running request-replica matching algorithm

Reduce the frequency of decision-making
Idea #1: Grouping requests by their external delays

- spatial coarsening of E2E decision-making
Idea #2: Reducing decision update frequency

- Temporal coarsening of E2E decision making
- Cache decision
- No need to compute the table per request

<table>
<thead>
<tr>
<th>External delay</th>
<th>Cached Decision</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;500ms</td>
<td>Replica_x</td>
</tr>
<tr>
<td>500-1200ms</td>
<td>Replica_y</td>
</tr>
<tr>
<td>&gt;1200ms</td>
<td>Replica_x</td>
</tr>
</tbody>
</table>
Evaluation

Set up
   Dataset: Real-world external delays from Microsoft traces
   Benchmark
      Default: Load balanced replica selection
      Idealized: Server-side delay is zero

Performance evaluation
   Overall performance
   E2E vs prior work
   E2E’s overhead
Overall performance of E2E

Microsoft trace: reshuffle the server-side delays vs default server-side delays
Distributed database: replica selection in Cassandra

![Graph showing QoE gain over default (%)]

- **QoE gain over default (%)**
  - Microsoft Trace
  - Cassandra

- **Legend**
  - E2E (Ours)
  - Idealized (Zero server-side delay)

August 22, 2019
E2E vs Prior work

E2E vs deadline-driven algorithm (Timecard [SOSP’13] )

Timecard: shortest-remaining time first

<table>
<thead>
<tr>
<th>Total delay deadline set by Timecard (sec.)</th>
<th>E2E</th>
<th>Timecard</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.0</td>
<td>15</td>
<td>10</td>
</tr>
<tr>
<td>3.4</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>5.9</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

QoE gain (%)
E2E’s overhead

- Machines in testbed: 3.0GHz Intel Xeon processor, 2GB RAM, 2GB RAM, 146G HDD and 1Gbps Ethernet link.

<table>
<thead>
<tr>
<th></th>
<th>Time consumption per request (ms)</th>
<th>Additional memory</th>
<th>Additional CPU</th>
<th>QoE gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>E2E (basic)</td>
<td>~100,000</td>
<td>~100%</td>
<td>&gt;100%</td>
<td>11.8%</td>
</tr>
<tr>
<td>E2E w/ grouping requests &amp; cache decision</td>
<td>~0.1</td>
<td>~7%</td>
<td>~2%</td>
<td>10.4%</td>
</tr>
</tbody>
</table>
Demo: How E2E works
Conclusion

Concurrent users have different sensitivities to server-side delays

Key idea: Embracing heterogenous user sensitivities leads to higher QoE

E2E: A concrete design to improve web QoE by allocating resource in accordance to user sensitivity

E2E improves QoE by up-to 15.4%, with negligible computing overhead

More details about E2E can be found in: https://people.cs.uchicago.edu/~zhangxu/e2e.html