Vroom: Accelerating the Mobile Web with Server-Aided Dependency Resolution

Vaspol Ruamviboonsuk\textsuperscript{1}, Ravi Netravali\textsuperscript{2}, Muhammed Uluyol\textsuperscript{1}, Harsha V. Madhyastha\textsuperscript{1}

\textsuperscript{1}University of Michigan, \textsuperscript{2}MIT
Mobile Web Dominant ... but Slow...

“9.85s to load median mobile retail sites” - Keynote Systems

“Average load time 14s on 4G” - DoubleClick

Problem: Slow web page loads

Mobile Optimized Popular Pages,
State of the Art Phone, Good LTE network
Simple Example Page

```html
<html>
  ...
  <script src="script.js"></script>
  ...
</html>
```

```javascript
var img = new Image();
img.src = "b.com/img.jpg";
document.body.appendChild(img);
```

Dependency Graph

- `index.html`
- `script.js`
- `img.jpg`
Loading a Web Page

Client

GET a.com

Parse HTML

HTML

Execute script.js

script.js

GET img.jpg

Onload

Dependency Graph

index.html

script.js

img.jpg

a.com

b.com
Waiting on CPU blocks network and vice versa

Key to fast page loads: Fully utilize CPU/network
Proxy Based Solution

GET a.com/index.html

Web servers must aid client in discovering resources

Shortcomings

- Client must trust HTTPS content pushed by proxy
- Proxy needs access to user’s cookies for all domains
Challenges to approach

1. How can web servers discover dependencies?

2. How do web servers inform clients of discovered dependencies?

3. How should clients use input from servers?
Challenges to approach

1. How can web servers discover dependencies?

2. How do web servers inform clients of discovered dependencies?

3. How should clients use input from servers?
Inefficient Page Load

Client

a.com

b.com

GET a.com

HTML

GET script.js

script.js

GET img.jpg

img.jpg

Parse HTML

Execute script.js

Onload

Dependency Graph

index.html → script.js → img.jpg
HTTP/2 Push

Client

GET a.com HTML

Parse HTML

Execute script.js

Onload

GET img.jpg

Dependency Graph

index.html → script.js → img.jpg
Push Only Load

Dependency Graph

index.html → script.js → img.jpg

Client

a.com

b.com

Onload

Parse HTML

Execute script.js

GET a.com

HTML

script.js

GET img.jpg

img.jpg

GET img.jpg
Dependency Hints

Client

a.com

b.com

GET img.jpg

HTML & b.com/img.jpg

Parse HTML

Execute script.js

Onload

index.html

script.js

img.jpg

Dependency Graph

GET a.com

GET b.com/img.jpg

GET img.jpg

GET img.jpg

GET img.jpg
Challenges to approach

1. **How can web servers discover dependencies?**

2. How do web servers inform clients of discovered dependencies?
   - **HTTP/2 Push + Dependency Hints**

3. How should clients use input from servers?
Strawman Dependency Resolution

Drawbacks

- Back-to-back loads differ
- Server cannot account for personalization
Combined Offline-Online Discovery

- **Stable dependencies**: Intersection of offline loads
- **Dynamic Content**: Online Parsing of HTML
Personalized Dependencies from Third-party Domains

- **a.com/index.html**
- **b.com/style.css**
- **b.com/logo_lo_res.png**
- **c.com/ad.html**
- **c.com/ad_inject.js**
- **d.com/car.gif**

**Start**

**Personalized Content?**
Challenges to approach

1. How do web servers discover dependencies?
   - Combine offline and online + Defer to third parties

2. How do web servers inform clients of discovered dependencies?
   - HTTP/2 Push + Dependency Hints

3. How do clients use input from servers?
Need for Scheduling

● No speedup with “Push All + Fetch ASAP”
  ○ Contention for access link bandwidth stalls processing

● Prioritize pushes and fetches of HTML, CSS, and JS
  ○ Schedule in order of processing
  ○ Account for 20% of bytes on average
Vroom scheduler in action

- **T=0**
  - Fetch all HTML, JS, CSS
  - Parse HTML and CSS, Execute JS

- Onload
  - Fetch other dependencies
Results overview

- **Accuracy of dependency discovery**
  - Median: 0% false positives and < 5% false negatives

- **Improvements in client perceived performance**
  - Speedup over status quo
  - Simple strawmанс don't suffice
  - Speedup even with warm caches
Results overview

- **Accuracy of dependency discovery**
  - Median: 0% false positives and < 5% false negatives

- **Improvements in client perceived performance**
  - *Speedup over status quo*
  - Simple strawmans don't suffice
  - Speedup even with warm caches
Evaluation Setup

Client

VPN over cellular network

Local desktop

HTTP/2 proxy

Apache server
Vroom makes page loads much faster

Alexa top 50 news and 50 sports sites
Vroom makes page loads much faster
Vroom makes page loads much faster
Vroom makes page loads much faster
Vroom makes page loads much faster

**Visual Improvement**
- Vroom also significantly reduces above-the-fold time

**Incremental Deployment**
- Most of Vroom’s benefits achievable with first-party adoption
Conclusion

● **Vroom: End-to-end solution that fully utilizes CPU/Network**

● Decouples dependency discovery from parsing and execution

● Decreases median page load time by 5s for popular sites