Polaris: Faster Page Loads Using Fine-grained Dependency Tracking

Ravi Netravali*   Ameesh Goyal*   James Mickens‡   Hari Balakrishnan*

*MIT CSAIL   ‡Harvard University
Web Performance

• Users demand fast page loads
• Slow page loads lead to lost revenue and low search rank

Research: Site Speed Is Hurting Your Everyone’s Revenue
IAN LURIE // MAY 9 2014
Site speed, site speed, site speed. Everyone around me is sick of hearing me talk about how slow my site is. Because I've pushed it on every client Portent's had since, oh, 2008.

Google Rank Website On Loading Time of the Page
By: Harsh Agrawal | In: SEO | Last Updated: 18/03/2015
This back Google webmaster team indicated that they will start ranking websites on their page loading time. Websites which take ages to load slows down the users and they are considering this factor seriously. Apart from other parameters like keywords, meta descriptions, Google will also consider Page load time as one of the reasons for your website search engine ranking.

How One Second Could Cost Amazon $1.6 Billion In Sales
Research on U.S. Net habits suggests that if this sentence takes longer than a second to load, many citizens will have clicked elsewhere already. If you've got the patience (or are European) read on for more shocking data on not dawdling.

It’s Official: Google Now Counts Site Speed As A Ranking Factor
Matt McGee on April 9, 2010 at 3:00 pm
Google has kept a promise it made last year: Site speed is now a ranking factor in Google’s algorithm, and is already in place for U.S. searchers. But Google also cautions web site owners not to sacrifice relevance in the name of faster web pages, and even says this new ranking factor will impact very few queries. More on that below, but first the background on today’s announcement from Google fellow Amit Singhal and Matt Cutts, head of Google’s web spam team.

Why Page Speed Matters
The first warning that site speed was on Google’s radar came last November, when Cutts said there...
Page Load

HTTP GET /index.html

HTTP GET /first.js

HTTP GET /second.js

HTTP GET /style.css

Client

x.com web server

4 RTTs
Dependency Graphs

Model page loads as directed acyclic graphs

- Page load time = time to completely resolve dependency graph

```html
index.html
<script src="x.com/first.js"/>
<script src="x.com/second.js"/>
<link src="x.com/style.css"/>

first.js
var x = 5;

second.js
var n = document.getElementsByTagName("link");
if ( n == 0 ) {...}

style.css
p {
  color: red;
}
```
Dependency Graphs

True Dependency Graph
3 RTTs

Lexical HTML Dependencies

<script src="x.com/first.js"/>
<script src="x.com/second.js"/>
<link src="x.com/style.css"/>

Conservative Assumptions

Missing Dependencies

index.html
first.js
second.js
style.css

HTML Tag Order
4 RTTs
Outline

• Scout: tracks fine-grained dependencies between page’s objects
  – Traditional dependency graphs *miss 30%* of edges

• Polaris: dynamic client-side scheduler written in JavaScript
  – Uses fine-grained dependencies to reduce page load times
    – 34% faster (*1.3 seconds*) on 12 Mbits/s link with 100 ms RTT
Scout

• Scout tracks many different dependencies across a page’s state

3 Types of Dependencies

<table>
<thead>
<tr>
<th>Write/Read</th>
<th>Read/Write</th>
<th>Write/Write</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>first.js</strong></td>
<td><strong>first.js</strong></td>
<td><strong>first.js</strong></td>
</tr>
<tr>
<td><code>x = 6;</code></td>
<td><code>x = [1,3,5];</code></td>
<td><code>alert(“first message”);</code></td>
</tr>
<tr>
<td><strong>second.js</strong></td>
<td><strong>second.js</strong></td>
<td><strong>second.js</strong></td>
</tr>
<tr>
<td><code>y = x + 5;</code></td>
<td><code>y = x.length;</code></td>
<td><code>alert(“second message”);</code></td>
</tr>
<tr>
<td></td>
<td><strong>third.js</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>x.push(7);</code></td>
<td></td>
</tr>
</tbody>
</table>
**Tracking Dependencies**

- **JS proxy objects**

  ```javascript
  var x = {'prop': 1};
  new Proxy({'prop': 1}, log_handlers);

  var y = x.prop;
  x.prop = 9;
  ```

- **Many others described in paper**
  - Global variables
  - Recursive proxying (e.g., x.y.z)
  - DOM (e.g., document.getElementById("foo"))
What we found

- Traditional graphs miss 30% of edges
- 80.8% of pages have altered critical paths

Adding all edges \(\rightarrow\) removes conservative assumptions \(\rightarrow\) faster page loads!
Polaris

Unmodified Web Browser

Client

HTTP(s) request (e.g., ‘GET /’)

HTTP(s) response

Scheduler Stub

Fine-grained Dependency Graph

Original HTML

Scheduler Logic

Web Servers

Offline Dependency Tracker (Scout)

Fine-grained Dependency Graph

<html> ...
</html>
Request Scheduling with Polaris

Always fetch objects on the dynamic critical path

- Loaded object
- Unloaded object

Diagram:
- Static critical path
- Dynamic critical path
Evaluating Polaris

- Gains increase with increasing RTT
- Gains increase with increasing link rate
- Baseline is Firefox (which does speculative execution)
- Large error bars: page structure matters too!
Impact of Dependency Graph Structure

- Apple: scheduling doesn’t matter (all requests have same priority)
- Weather: short and long chains so scheduling matters
Conclusion

• Browsers today are constrained by uncertainty!
  – Conservative assumptions lead to higher page load times
  – Klotski (NSDI ’15), WProf (NSDI ‘13), Browser Developer Tools

• Scout: tracks fine-grained dependencies between page’s objects
  – Prior dependency graphs miss 30% of edges

• Polaris: dynamic client-side scheduler written in JavaScript
  – Uses Scout’s fine-grained dependencies to reduce page load times
  – 34% faster (1.3 seconds) on 12 Mbits/s link with 100 ms RTT

http://web.mit.edu/polaris  polarisweb@mit.edu