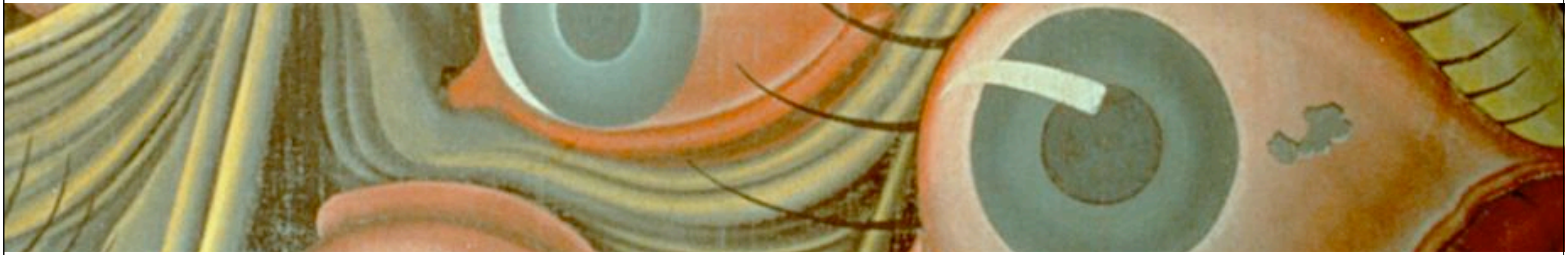
Jan Vitek

Orthodoxy

Static is Better

how *dynamic* is dynamic?

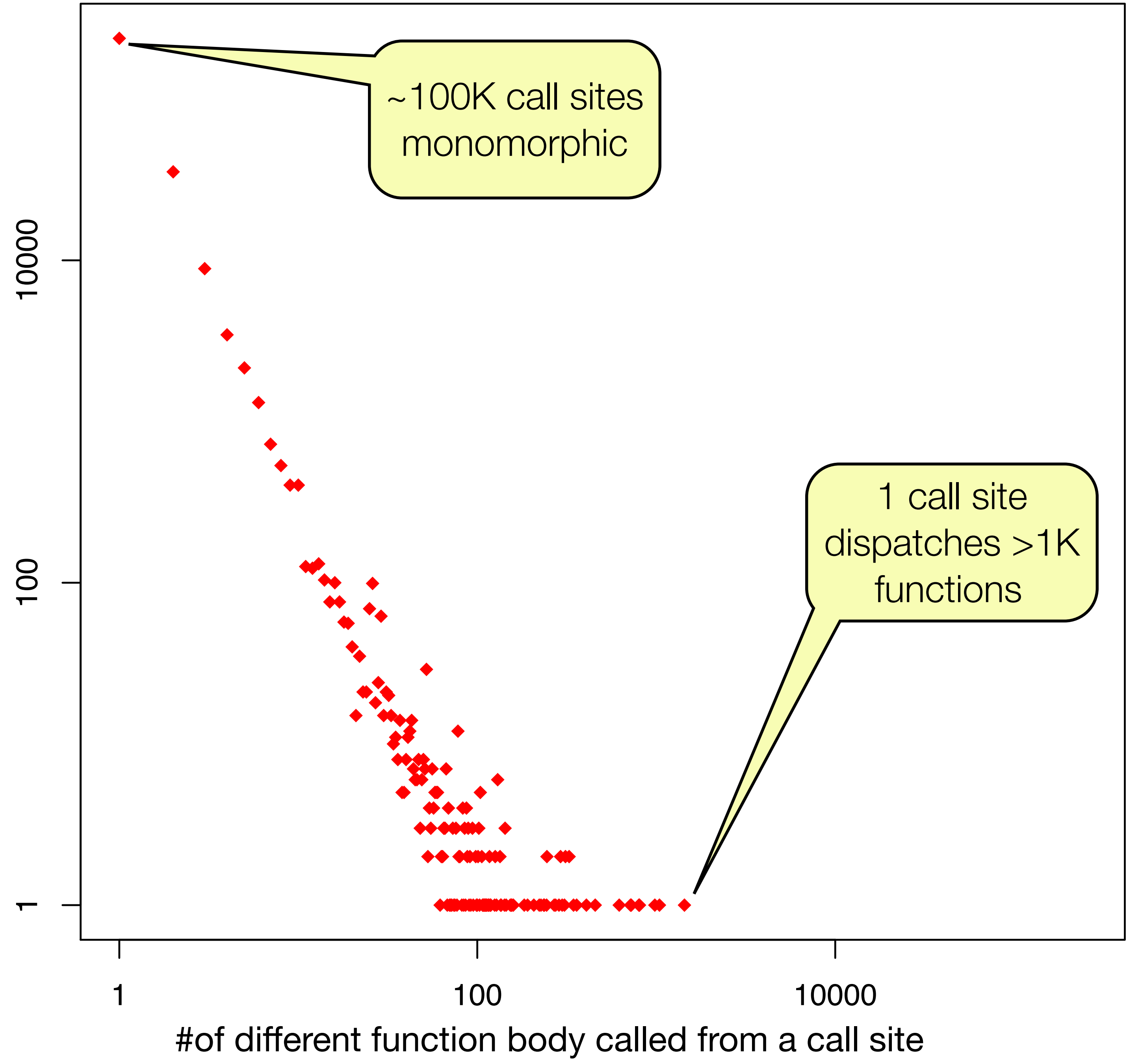


Richards, Lesbrene, Burg, Vitek. **An Analysis fo the Dynamic Behavior of JavaScript Programs.** PLDI'10

assumptions

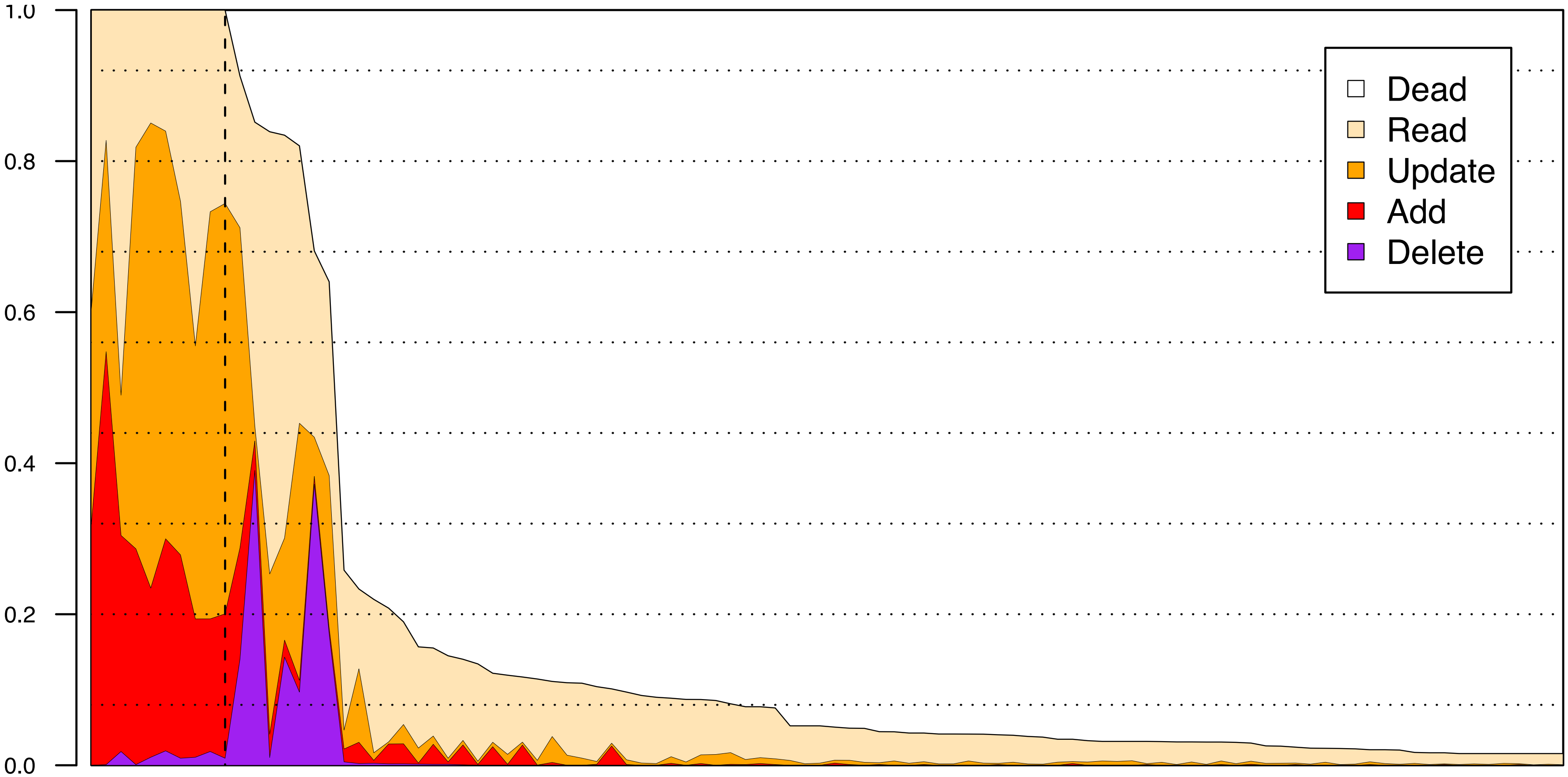
1. Call-site Dynamism is Low
2. Properties are Added at Object Initialization
3. Properties are Rarely Deleted
4. `eval` is Infrequent and Harmless
5. ...

Call-site Dynamism is Low


~100K call sites monomorphic
1 call site dispatches >1K functions
1
100
10000
1
100
10000
#of different function body called from a call site

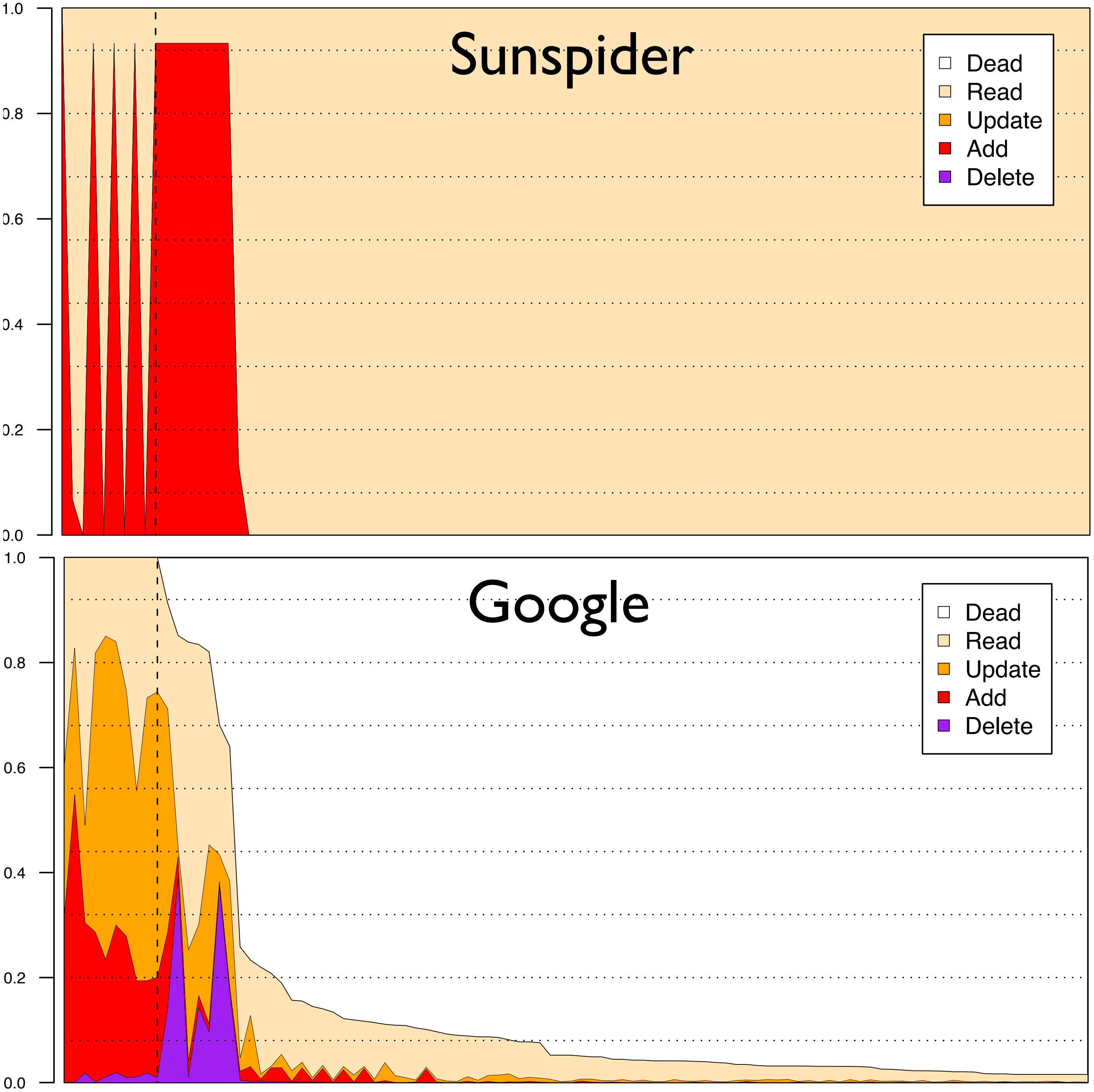
Properties are Added at Object Initialization

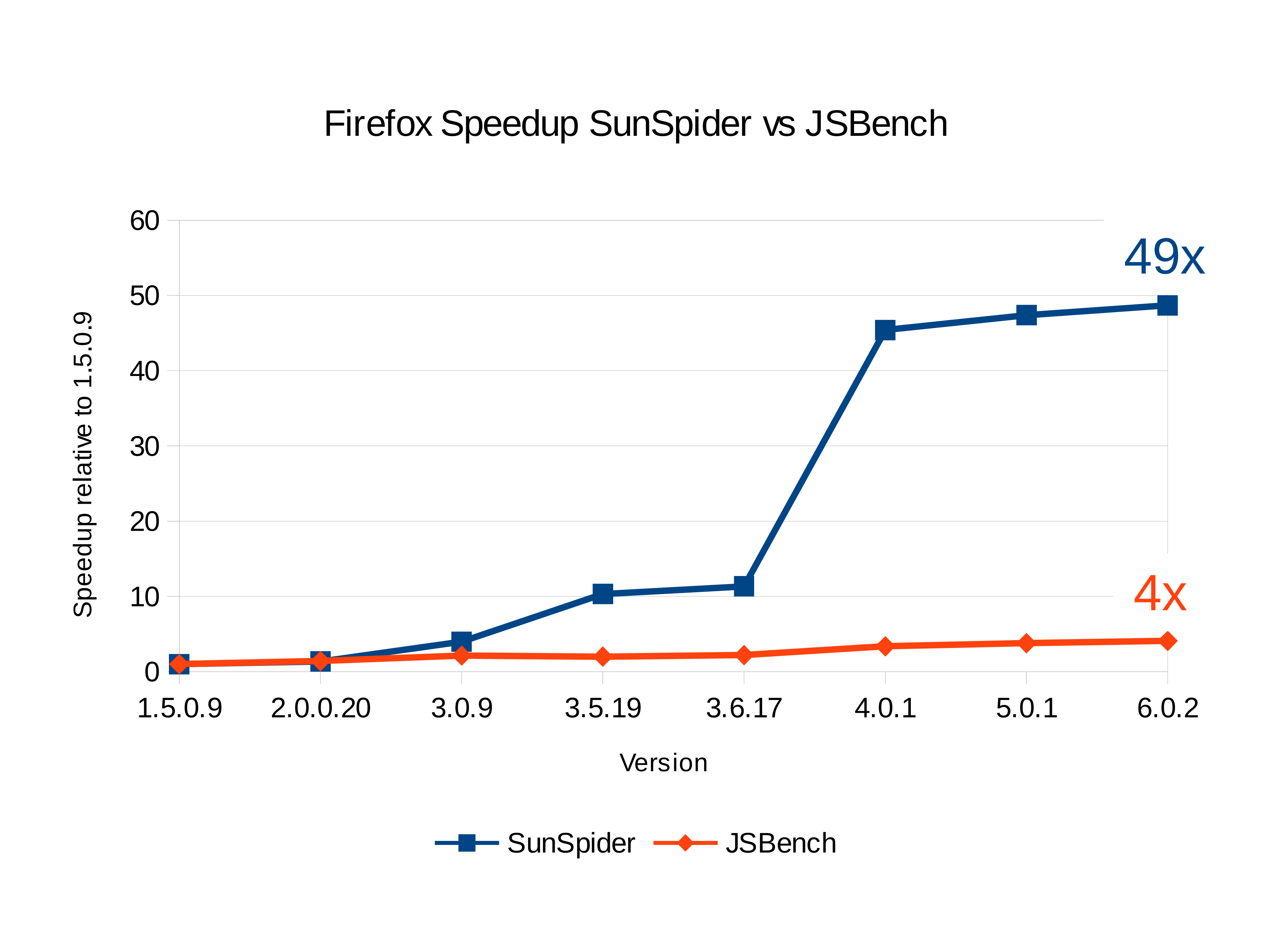
Google



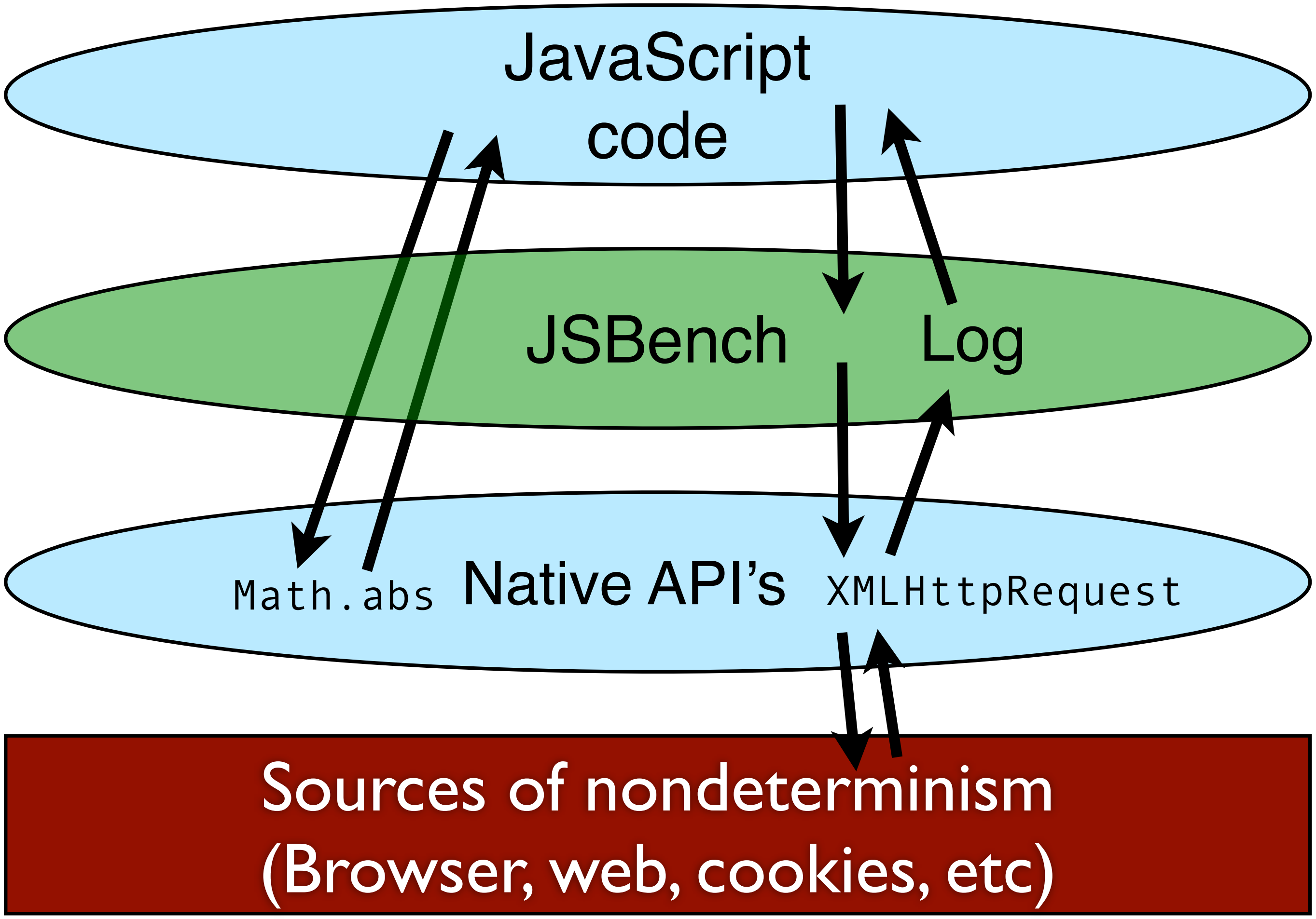
benchmarks for free

Richards, Gal, Eich, Vitek. **JSBench: Automating the Construction of JavaScript Benchmarks.** OOPSLA'11


Sunspider
1.0
0.8
0.6
0.4
0.2
0.0
□ Dead
□ Read
🟧 Update
🟥 Add
🟪 Delete
Google
1.0
0.8
0.6
0.4
0.2
0.0
□ Dead
□ Read
🟧 Update
🟥 Add
🟪 Delete


Firefox Speedup SunSpider vs JSBench
60
50
40
30
20
10
0
Speedup relative to 1.5.0.9
49x
4x
1.5.0.9
2.0.0.20
3.0.9
3.5.19
3.6.17
4.0.1
5.0.1
6.0.2
Version
SunSpider
JSBench

Record



JavaScript code

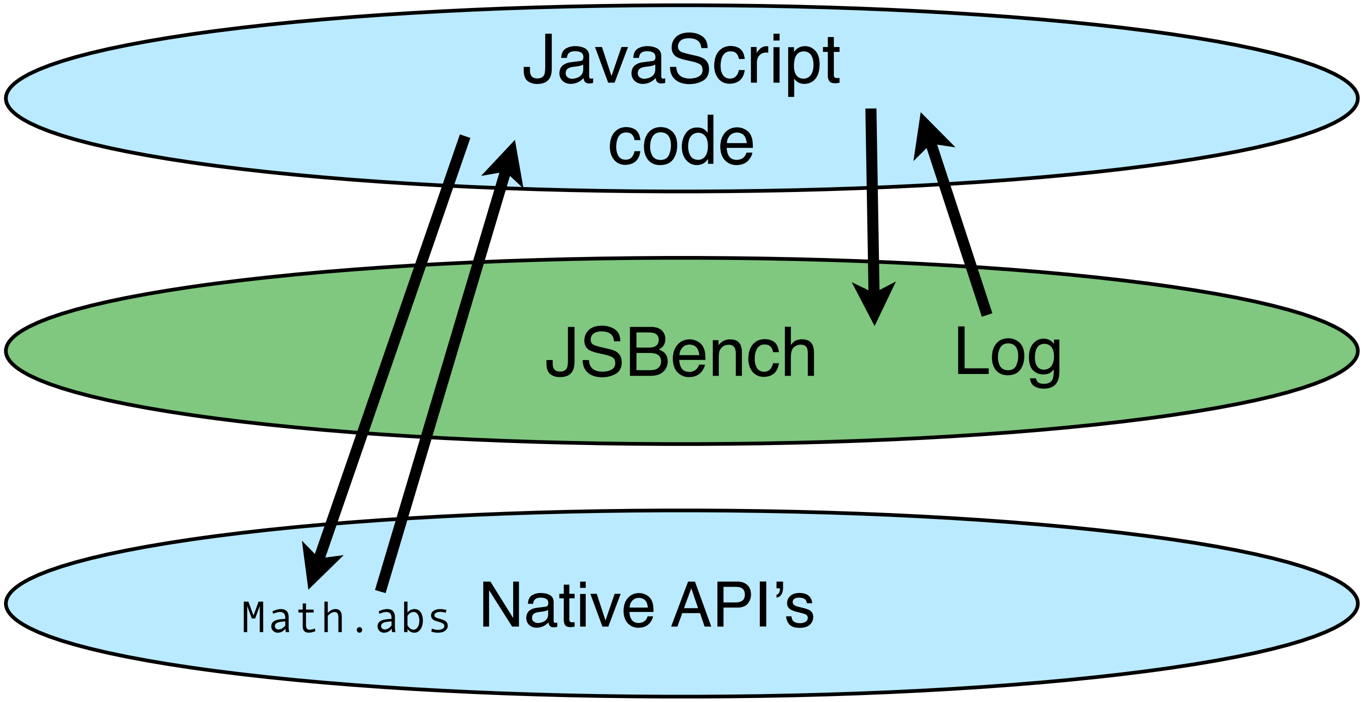
JSBench

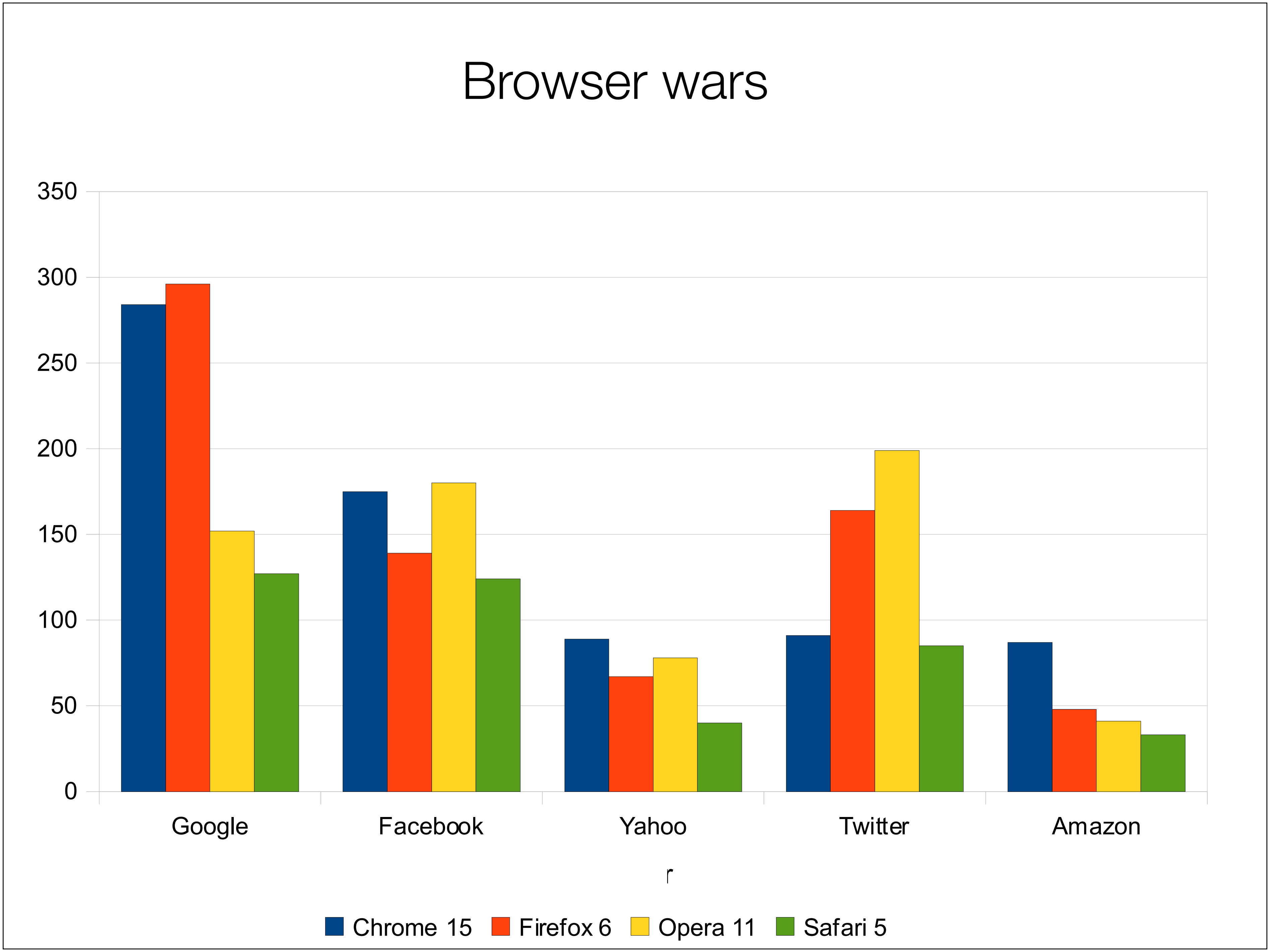
Log

`Math.abs` Native API's `XMLHttpRequest`

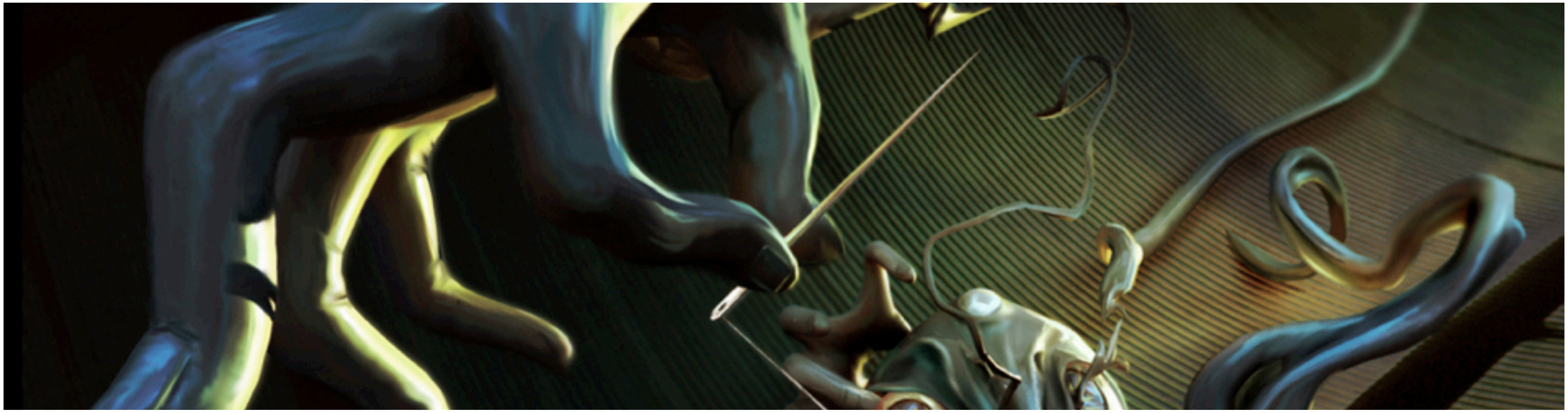
Sources of nondeterminism
(Browser, web, cookies, etc)

Replay




Browser wars
350
300
250
200
150
100
50
0
Google
Facebook
Yahoo
Twitter
Amazon
r
Chrome 15
Firefox 6
Opera 11
Safari 5

looking for the mythical eval



Richards, Hammer, Burg, Vitek. **The Eval that Men Do: A Large-scale Study of the Use of Eval in JavaScript Applications.** ECOOP 2011

A Flash of Eval

```
 var flashVersion = parse();
flash2Installed = flashVersion == 2;
flash3Installed = flashVersion == 3;
flash4Installed = flashVersion == 4;
flash5Installed = flashVersion == 5;
flash6Installed = flashVersion == 6;
flash7Installed = flashVersion == 7;
flash8Installed = flashVersion == 8;
flash9Installed = flashVersion == 9;
flash10Installed = flashVersion == 10;
flash11Installed = flashVersion == 11;
for (var i = 2; i <= maxVersion; i++)
 if(eval(”flash”+i+”Installed”)==true)
   actualVersion = i;
```

Corpus

- Top 10,000 web sites (from Alexa.com)

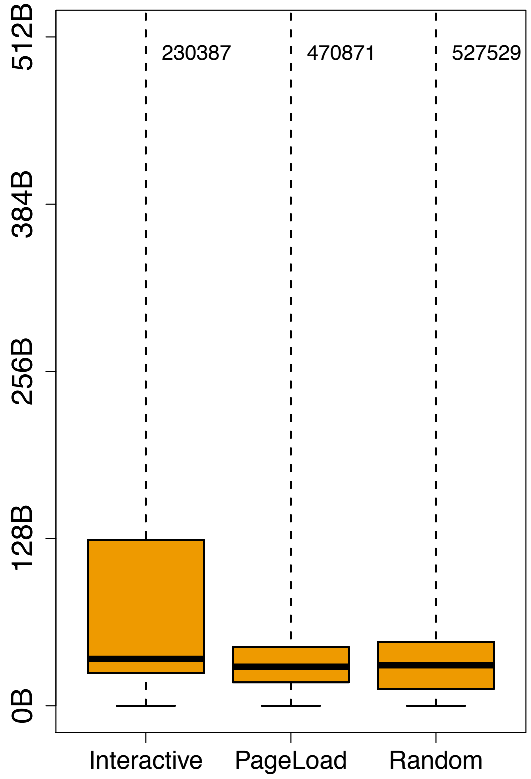
3,346MB JavaScript, **337MB** of eval strings, **550,358** calls

100% of top 100 sites use JavaScript

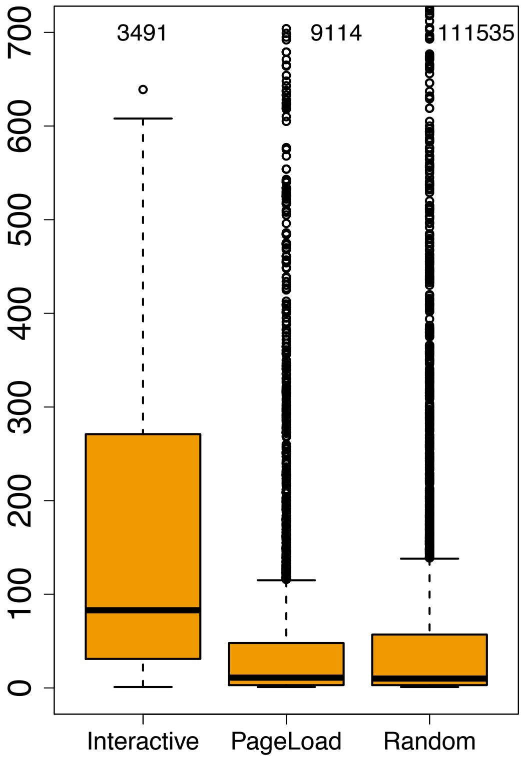
Eval Usage

82% use eval!

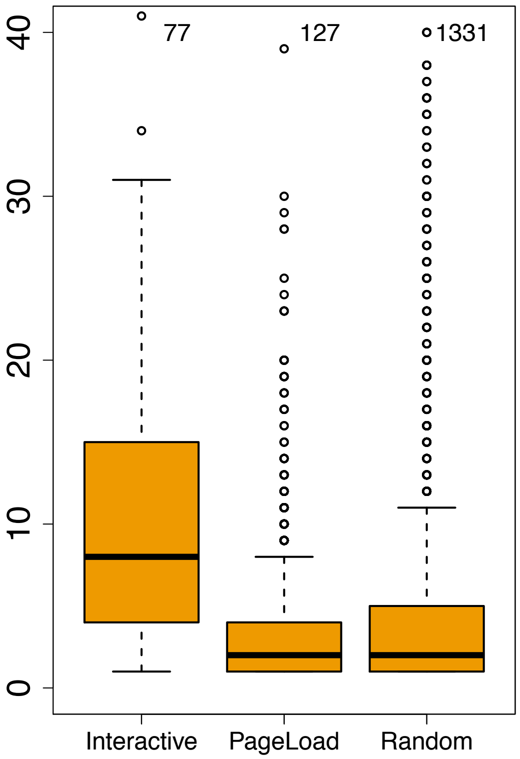
String Size



Calls



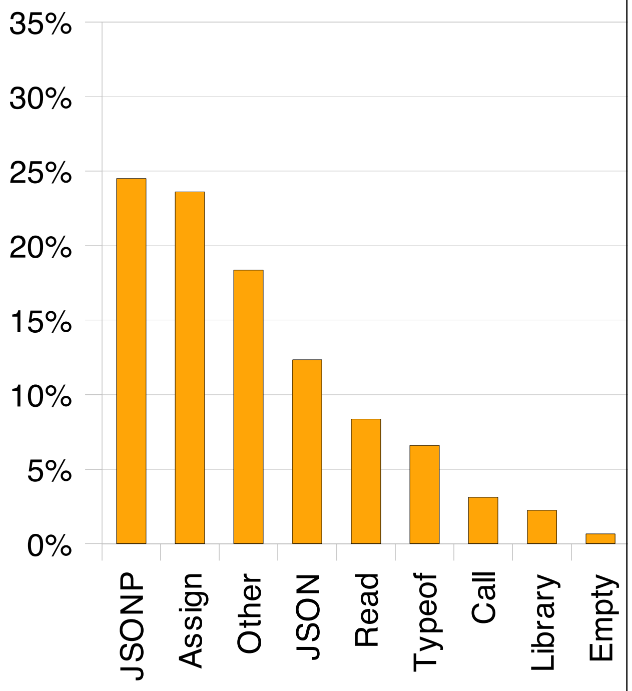
Call Sites



The Shape of Eval

Identified common patterns:

JSON	eval(‘{“x”: 2}’)
JSONP	eval(“f({x: 2})”)
Library	
Read	eval(“obj . f”)
Assign	eval(“id = x”)
Typeof	eval(‘typeof(’+x+’)!=”undefined”’)
Try	eval(‘try{throw v=14}catch(e){}’)
Call	eval(‘get(”menu”)’)
Empty	
(Other)	

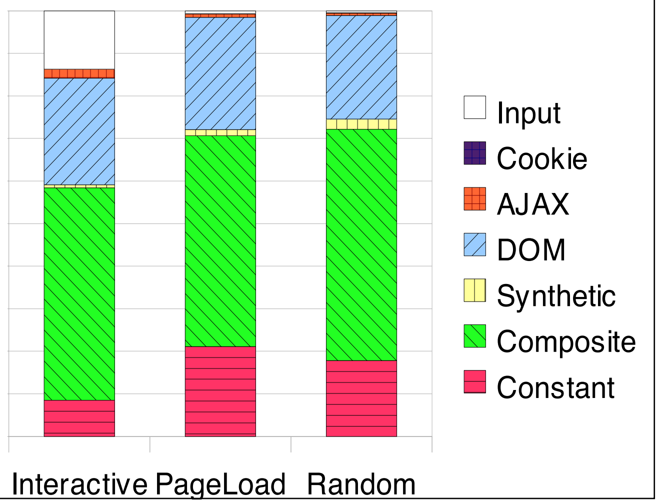


(a) INTERACTIVE

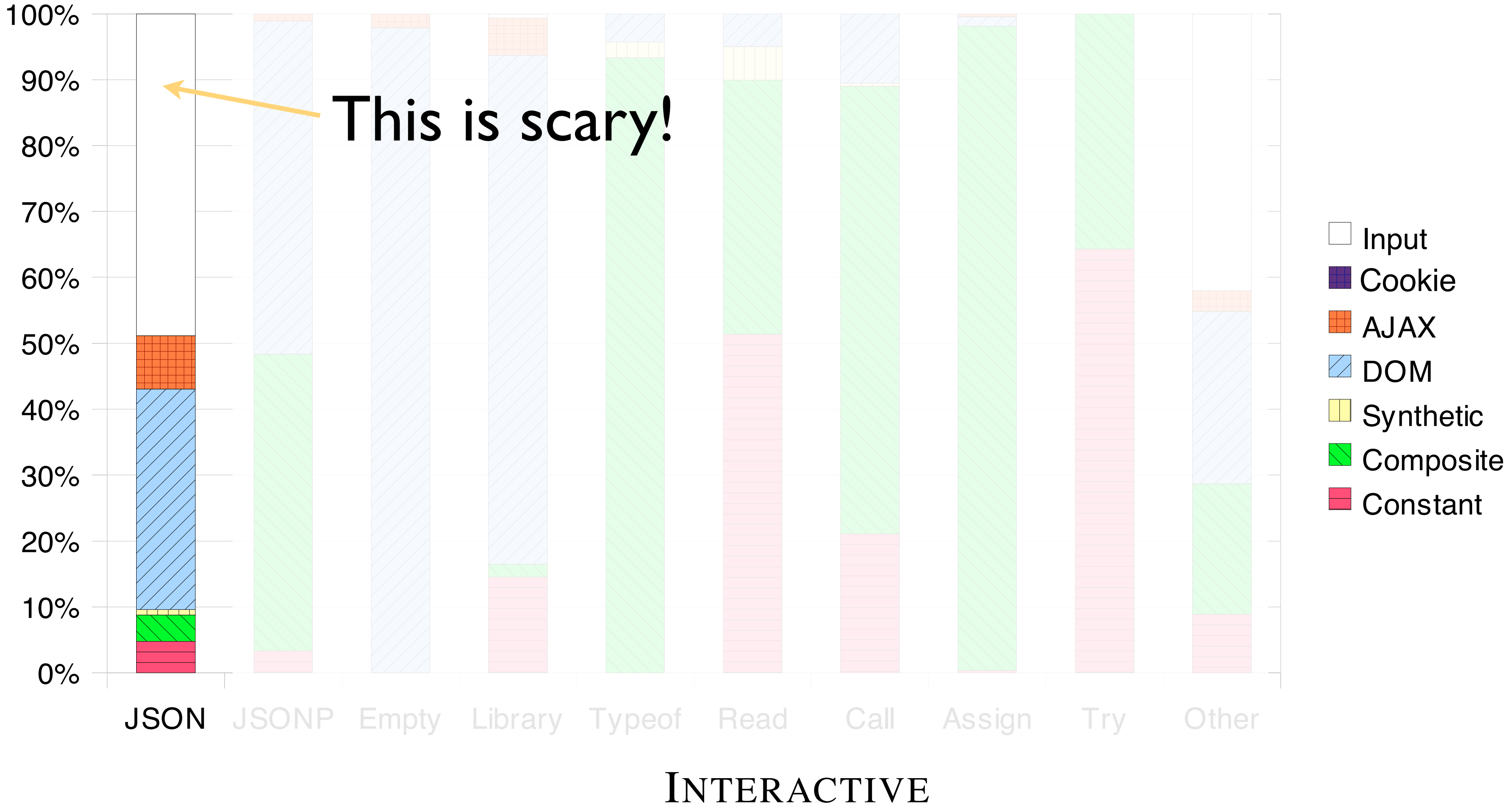
Provenance of eval strings:

Constant	eval(“x”)
Composite	eval(x+“y”)
Synthetic	eval(“eval(‘”+x+“’)”)
DOM	eval(document.getById(“x”).text)
AJAX	eval(xmlhttprequest.responseText)
Cookies	eval(document.cookie.substr(...))
Input	eval(document.getById(“username”).value)

The Root of Eval

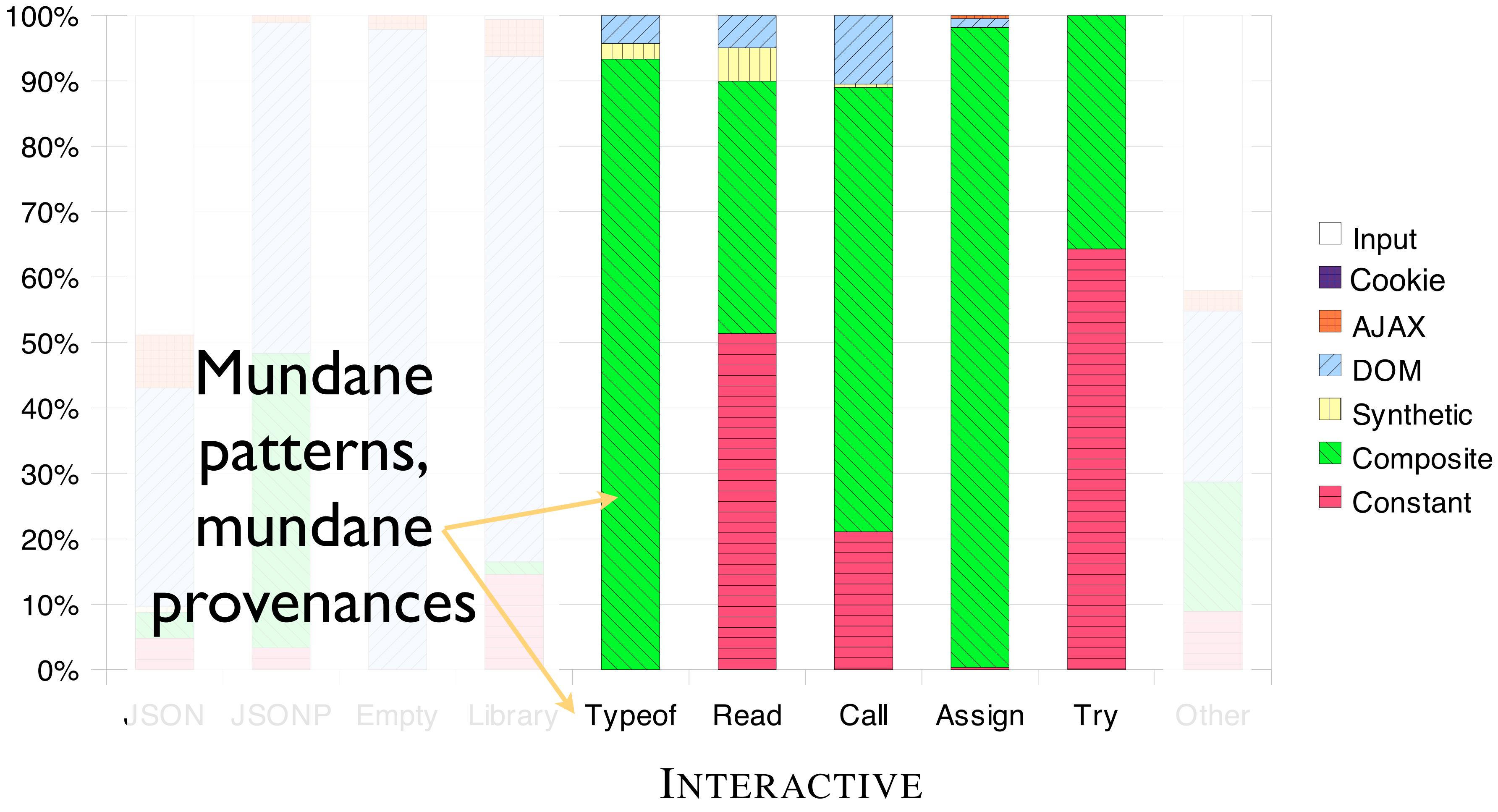


Provenance v Patterns



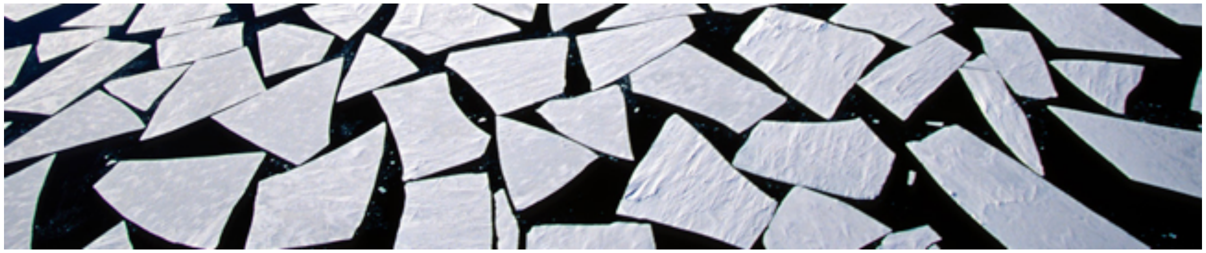
This is scary!

Provenance v Patterns

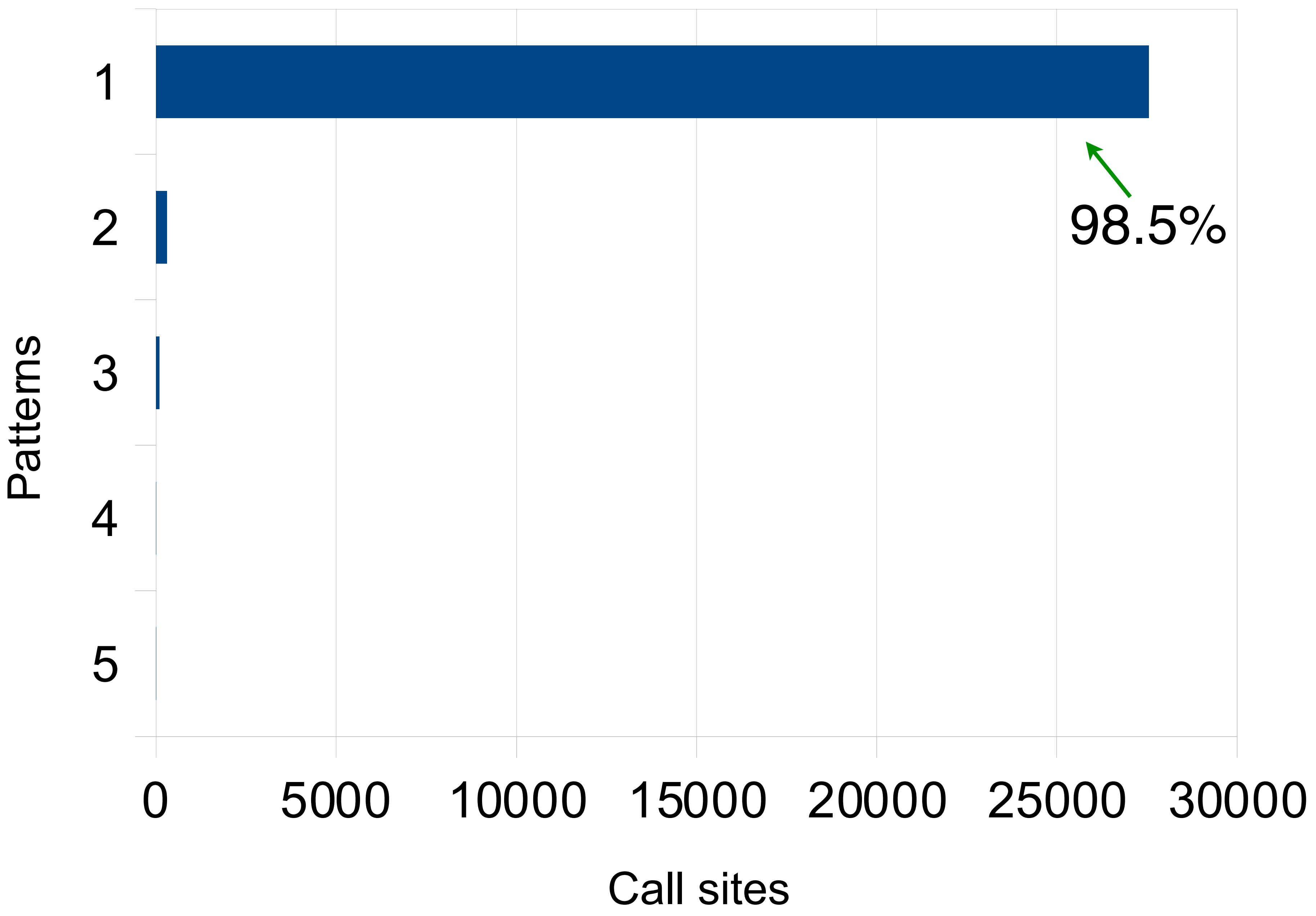


Mundane patterns, mundane provenances

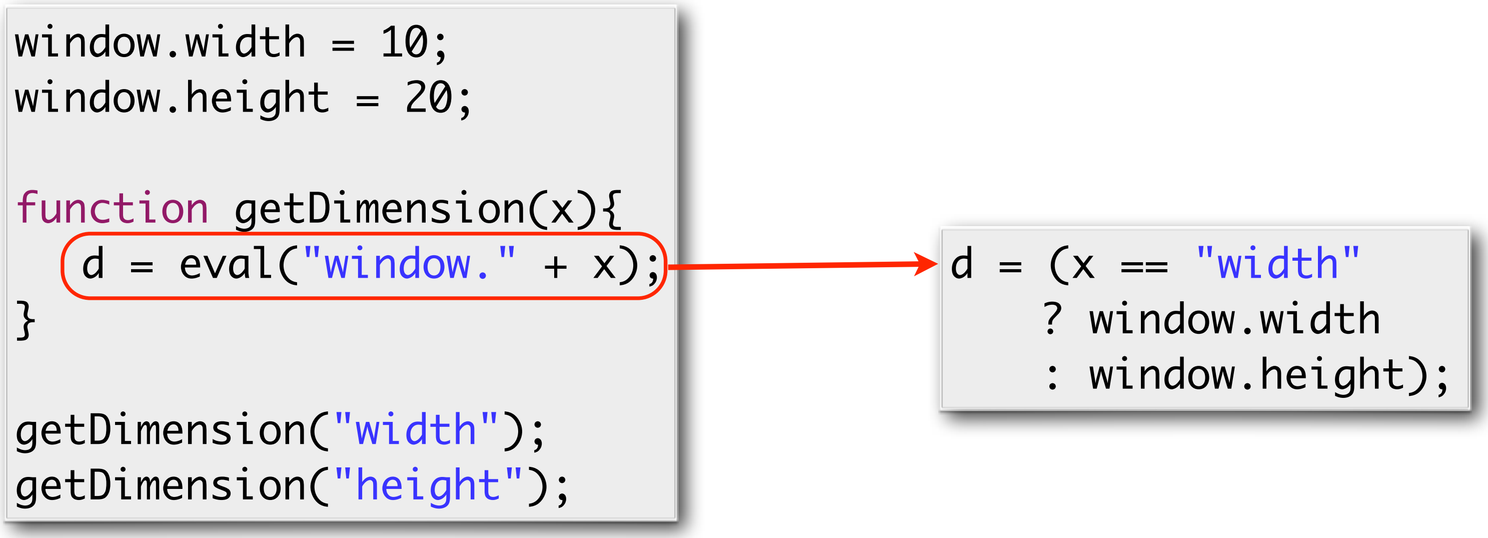
eval begone!



Meawad, Richards, Morandat, Vitek. **Eval Begone! : Semi-Automated Removal of Eval from JavaScript Programs** OOPSLA '12


1
2
3
4
5
Patterns
98.5%
0
5000
10000
15000
20000
25000
30000
Call sites

Example classifier

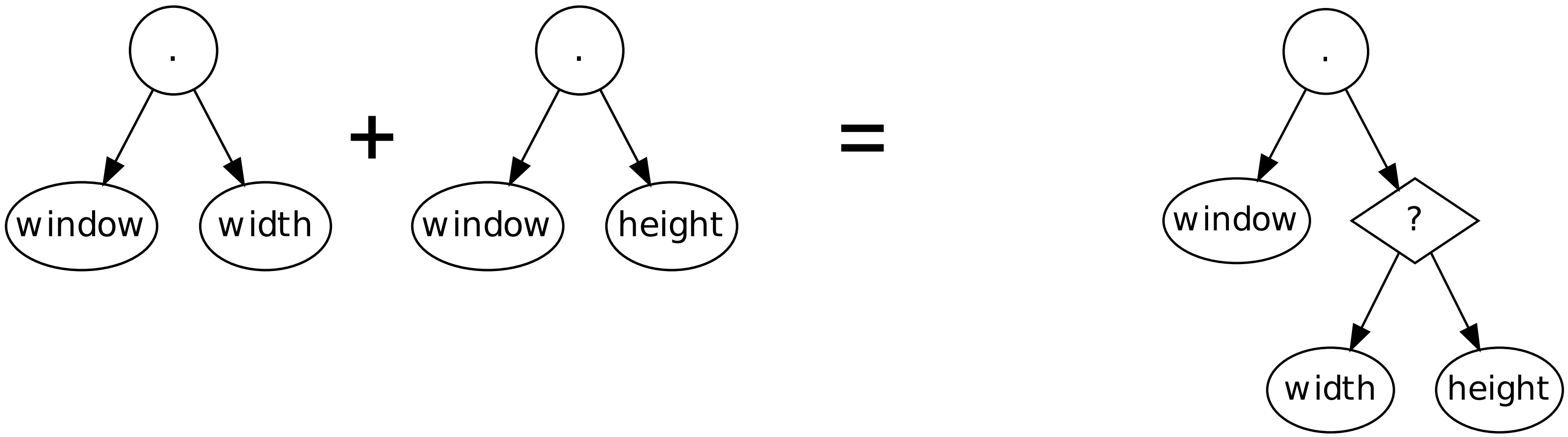


```
window.width = 10;
window.height = 20;

function getDimension(x){
   d = eval("window." + x);
}

getDimension("width");
getDimension("height");
```

```
d = (x == "width"
     ? window.width
     : window.height);
```



Validation

Once we’ve generated a classifier, can it accept new input?

- Evals from interactive use of top 100 web pages
- Train on *k* strings, test on remainder
- With *k*≥3, 95% of sites with no misprediction

Planet Dynamic

or: How I Learned to Stop Worrying and Love Reflection