a. A brief introduction of raw experimental data generated (copy the raw data from the console and explain it)

On a high level the experiment does 1 and 2 and then compares the result.

1. Run experiment by disabling TFO
2. Run experiment by enabling TFO

Console data for 1 & 2 show TFO status as below:

TFO: 0, TFO: 3

Within each step, run the experiment with 3 different network delay time 10ms, 50ms, 100ms and for each network delay value, there are 2 test runs. In each test run, each url is fetched once and the PLT is The network is fetched beforehand and the experiment simulates the real work by controlling the network delay.

For example, when there are 2 urls specified in myURLS.pages and TFO is disabled and the network delay time is 100ms, the console output is:

Experiment 0 http://b-ok.org @ 100ms delay
Fetching http://b-ok.org took 22.050s.
Experiment 0 http://canvas.gatech.edu @ 100ms delay
Fetching http://canvas.gatech.edu took 31.988s.
Experiment 1 http://b-ok.org @ 100ms delay
Fetching http://b-ok.org took 3.866s.
Experiment 1 http://canvas.gatech.edu @ 100ms delay
Fetching http://canvas.gatech.edu took 10.171s.
Experiment done

The average PLT of experiment 0 and 1 is taken for TFO enabled and disabled for each url respectively. In this example, the average PLT for TFO disabled is 27.019s for 21.08s for canvas.gatech.edu.

Same experiment is run for TFO enabled scenario and console output is something like:

Experiment 0 http://b-ok.org @ 100ms delay
Fetching http://b-ok.org took 4.039s.
Experiment 0 http://canvas.gatech.edu @ 100ms delay
Fetching http://canvas.gatech.edu took 8.893s.
Experiment 1 http://b-ok.org @ 100ms delay
Fetching http://b-ok.org took 2.535s.
Experiment 1 http://canvas.gatech.edu @ 100ms delay
Fetching http://canvas.gatech.edu took 9.157s.

We can calculate the average PLT similarly.

And the improvement for enabling TFO = PLT for disabled / PLT for enabled - 1.

b. Your analysis should answer the following questions for each URL you specified:

canvas.gatech.edu

Page	RTT(ms)	PLT: no TFO (ms)	PLT: TFO (ms)	Improv.
httpcanvas.gatech.edu				
	200	21907.403	8672.016	60.41513456
	100	7226.638	6325.094	12.47528934
	20	6363.627	6168.719	3.062844507

TFO reduces the page load time, especially significant for 100 and 200 RTT experiment.

How does the RTT value affect these results?

As TFO improves the PLT by eliminating one RTT of extra latency. The higher the RTT, the bigger the improvement by TFO. The impact on 20 ms RTT is only 3% (compared to that of 12% and 60% for higher RTTs) is because the bottleneck is on analyzing content of the website instead of network delay for small RTT.

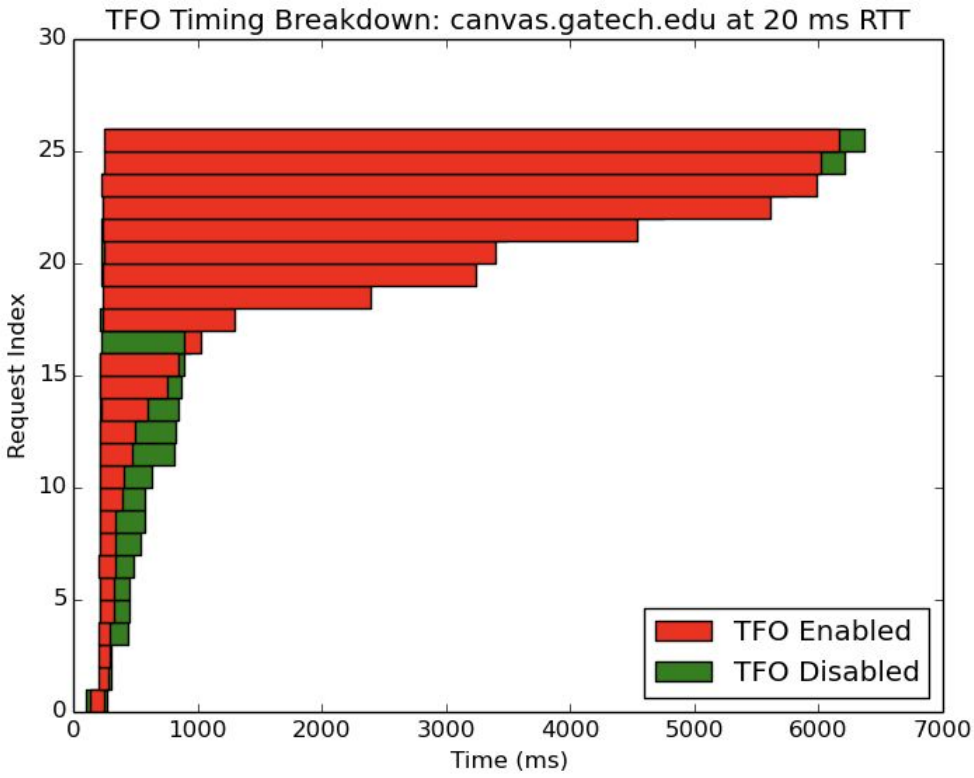
Does the particular content available at this URL lend itself to performance enhancements provided by TFO?

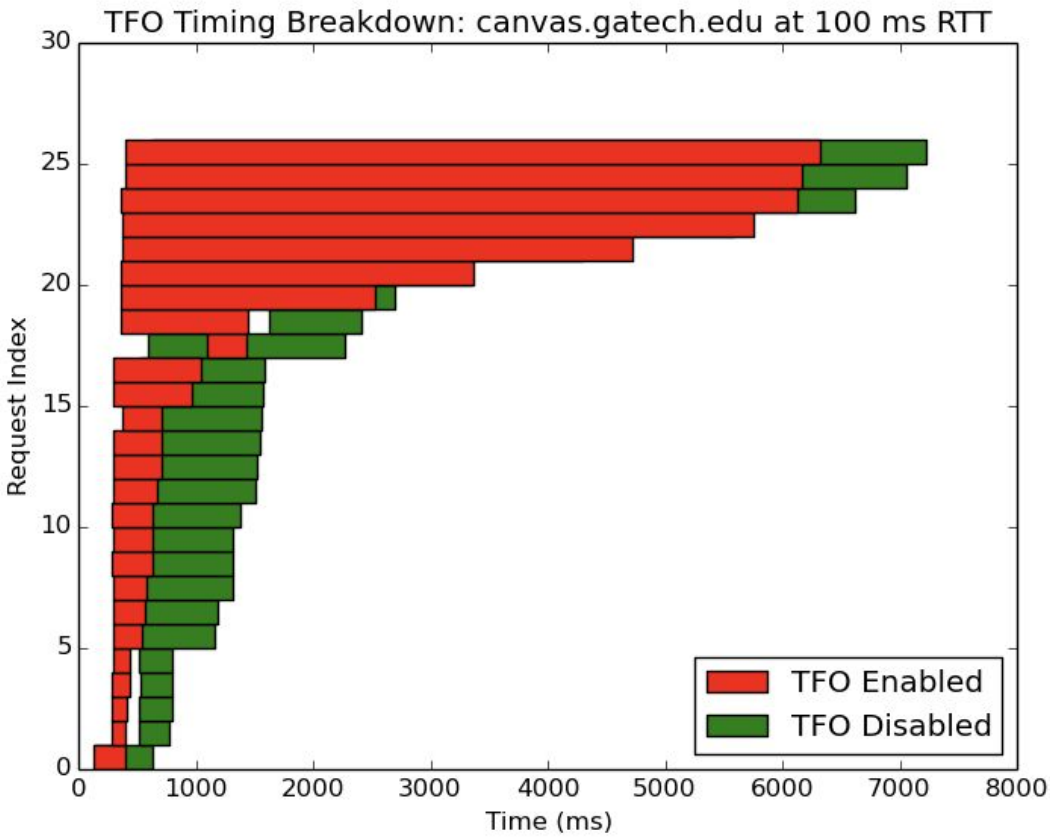
Major contents are images, which are less adaptive to improvement from TFO.

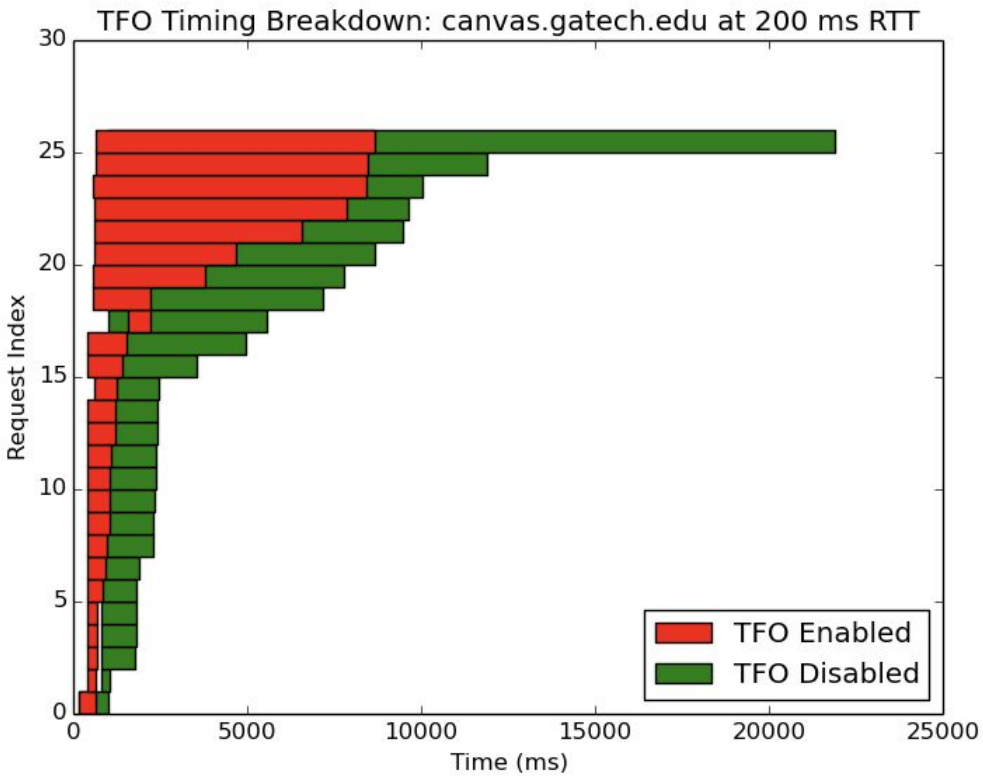
Were these results surprising in any way?

For each index, the improvement seems to come more from the increased PLT with Disabled TFO, rather than the decreased PLT with Enabled TFO.

Include relevant graphs from the output-figures folder


TFO Timing Breakdown: canvas.gatech.edu at 20 ms RTT
Request Index
Time (ms)
TFO Enabled
TFO Disabled
0
5
10
15
20
25
30
0
1000
2000
3000
4000
5000
6000
7000


TFO Timing Breakdown: canvas.gatech.edu at 100 ms RTT
Request Index
Time (ms)
0
5
10
15
20
25
30
0
1000
2000
3000
4000
5000
6000
7000
8000
TFO Enabled
TFO Disabled



b-ok.org

What effect does TFO have on the timing?

Page	RTT(ms)	PLT: no TFO (ms)	PLT: TFO (ms)	Improv.
httpb-ok.org				
	200	21966.276	2459.238	88.80448375
	100	2631.534	1433.924	45.50995731
	20	1180.104	980.791	16.88944364

There exist very significant improvement on PLT. Ranging between 16% to 89% for the RTT tested on.

How does the RTT value affect these results?

Similar to the reason as for canvas.gatech.edu, the higher the RTT, the bigger the improvement by TFO.

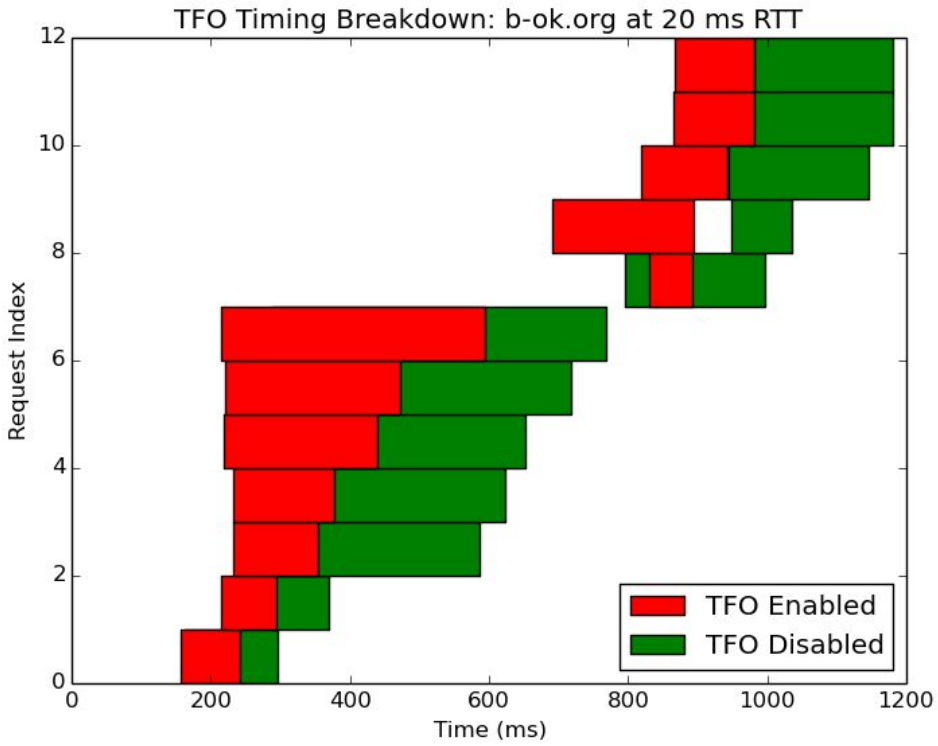
Does the particular content available at this URL lend itself to performance enhancements provided by TFO?

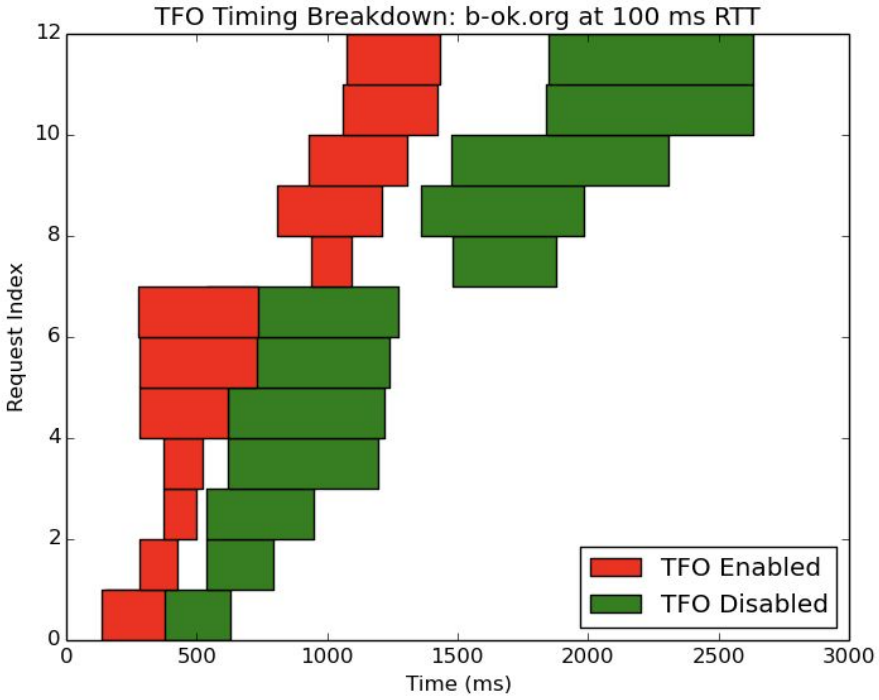
This is a very simple website. The content is about ⅓ the size of canvas and are much more simpler than the images in canvas. Elements such as simple text and small images see higher percentage of improvements.

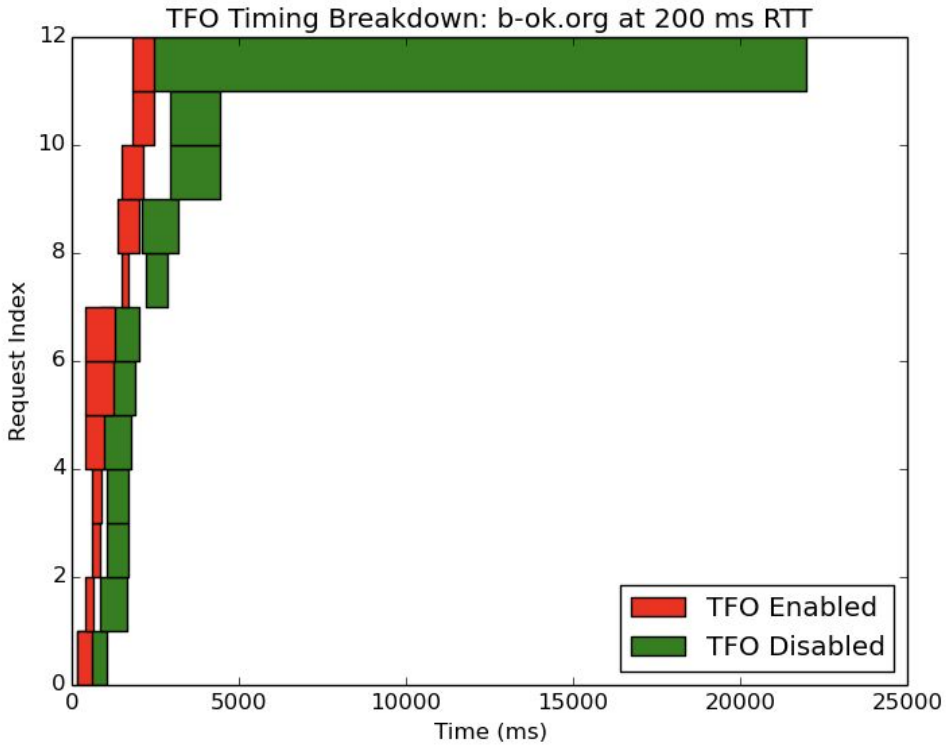
Were these results surprising in any way?

200 ms TFO Disabled PLT increase way more than that the increase in RTT for certain request.

Include relevant graphs from the output-figures folder







bbc.com

What effect does TFO have on the timing?

Page RTT(ms) PLT: no TFO (ms) PLT: TFO (ms) Improv.

httpwww.bbc.com

200 32727.442 5000.083 84.7220476321

100 2114.017 1184.372 43.975284967

20 733.767 702.862 4.21182746022

The improvement is similar to that of canvas on the smaller RTT. But for bigger RTT, seeing similar improvement as b-ok.org.

How does the RTT value affect these results?

Similar to the reason as for canvas.gatech.edu, the higher the RTT, the bigger the improvement by TFO.

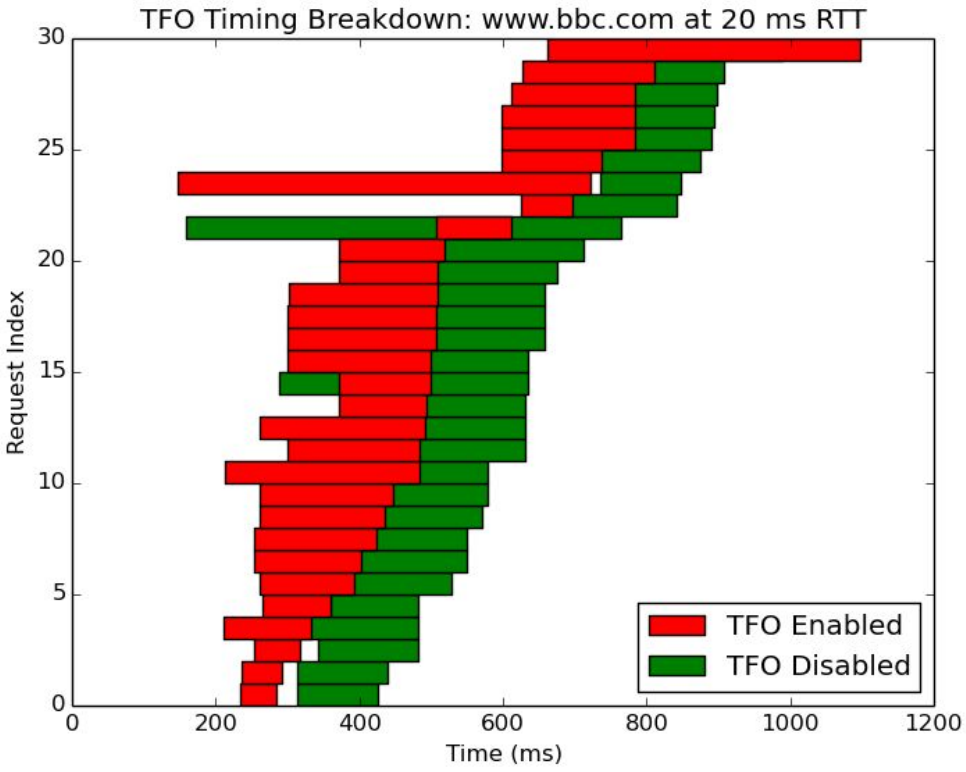
Does the particular content available at this URL lend itself to performance enhancements provided by TFO?

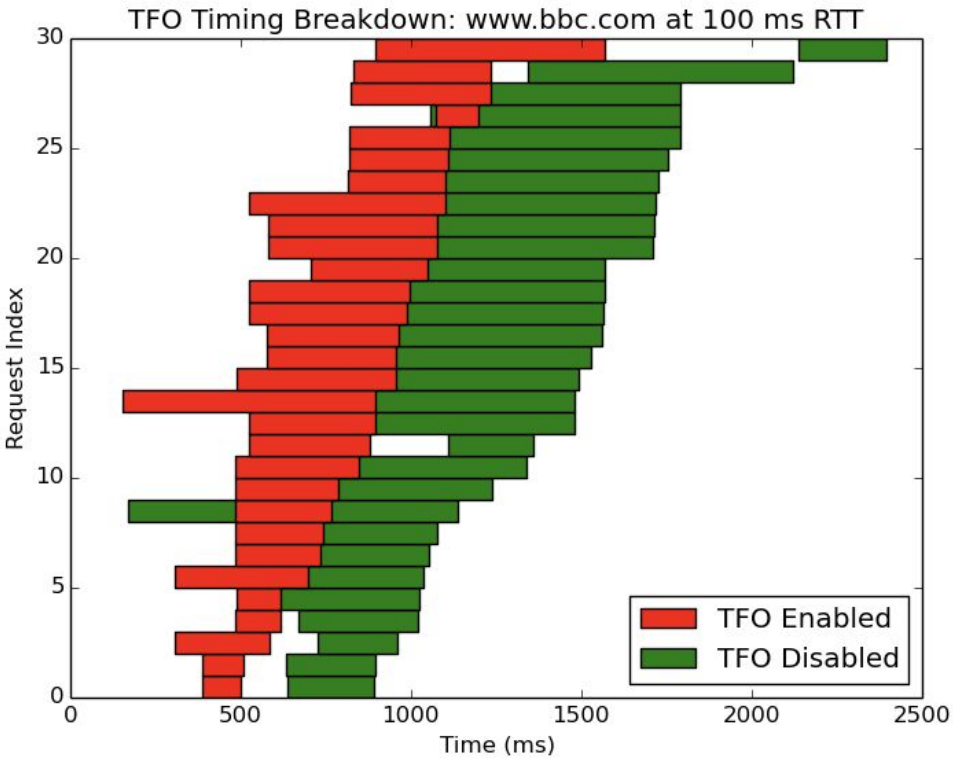
Similar to that of canvas, but bbc has even more images, so even less adaptive to the TFO performance improvements.

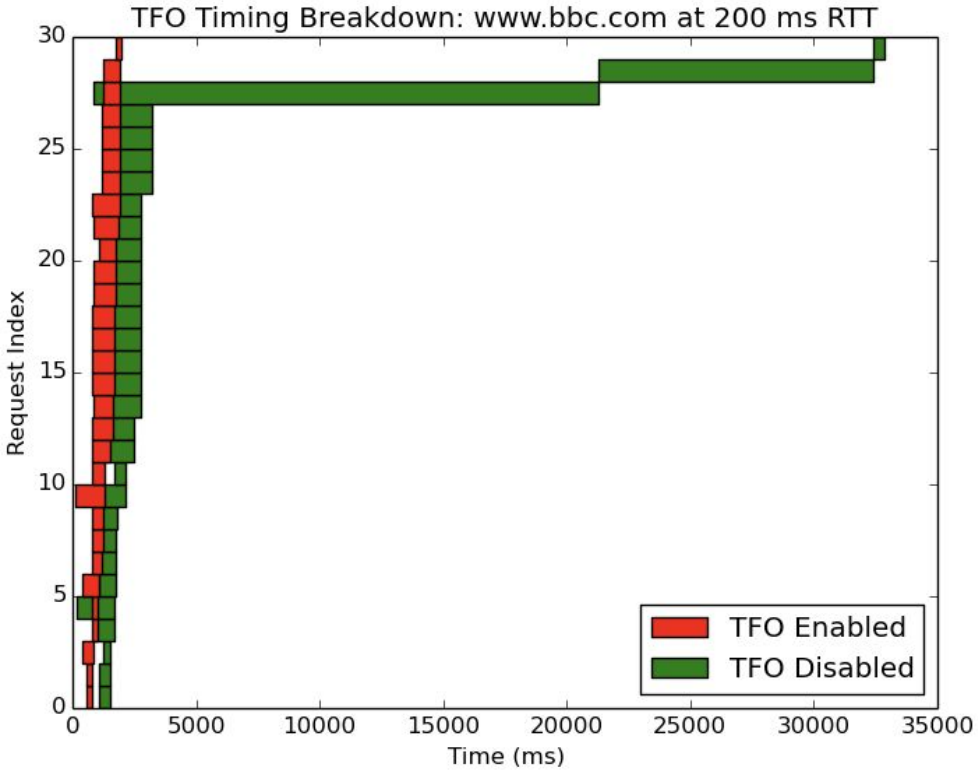
Were these results surprising in any way?

200 ms TFO Disabled PLT increase way more than that the increase in RTT for certain request.

Include relevant graphs from the output-figures folder




TFO Timing Breakdown: www.bbc.com at 100 ms RTT
Request Index
Time (ms)
TFO Enabled
TFO Disabled
0
5
10
15
20
25
30
0
500
1000
1500
2000
2500


TFO Timing Breakdown: www.bbc.com at 200 ms RTT
Request Index
Time (ms)
TFO Enabled
TFO Disabled
0
5
10
15
20
25
30
0
5000
10000
15000
20000
25000
30000
35000

c. Include a brief summary of your findings and state what conclusions you can draw based on the results of your experiment.

   TFO improves PLT for all the experiments ran and for most of the request index.

   Emulated RTT smaller => More improvement expected by TFO as the network delay is a small fraction of the PLT and the resource processing is not taking a relatively large time. So the smaller gain is expected from TFO.

   Page heavier in content => Bottleneck is on analyzing the content instead of network delay => less improvement by TFO.

d. Based on the reading and your experiment, where do you see TCP Fast Open having the best potential for improvement? What about the worst?

   Best potential: Improvement on simple page with less content and on mobile apps, which usually have higher RTT.

   Worst potential: Devices with relatively less network delay or contents are heavy in content. RTT doesn't really become the bottleneck so TCP won't play a big role here.