# TTI Variability study

Author: dproy@
Created: 2 March 2017
Last updated: 2 March 2017

*TL;DR: Look at these graphs.*

## Experiment

We ran the networkReverseSearch definition of firstInteractive against 121 sites with 25 runs each. The experiment was run on live sites, with Regular-3G traffic shaping, on Nexus 5X android phones in cluster telemetry.

## Results

For 71 of these sites , time to first interactive was very stable. For 38 of these sites, TTI was multi-modal, or just had too much variance in general. We did not have enough data for 12 of these sites, probably because we did not wait long enough for the site to reach TTI consistently.

We classified variability of TTI for each site as good or bad by eyeballing the graph - there is no particular threshold of standard deviation or relative standard deviation that was used. We found none of these numbers captured the variability behavior effectively enough, and it took less than five minutes to label all the graphs as good/bad, which is probably less time than it would have taken us to find a good meta metric for expressing variability.

## Recommendation

The sites that had too much variance can be grouped into two categories:

- Sites that showed clear bimodal/trimodal behavior (e.g. vevo case below) because of our window size / long task threshold
- Sites that showed too much variance in TTI because it executes too much javascript and all the variance in task execution time adds up (e.g. silikonvadisi case below), or the site executes differently each time it is loaded (e.g. part of the variance in yahoo case below)

There is no way to fix TTI in case 2, except may be by tightly controlling the environment - for example using recorded network traffic and perhaps using a more deterministic version of chrome. We can do neither of this in UMA context.

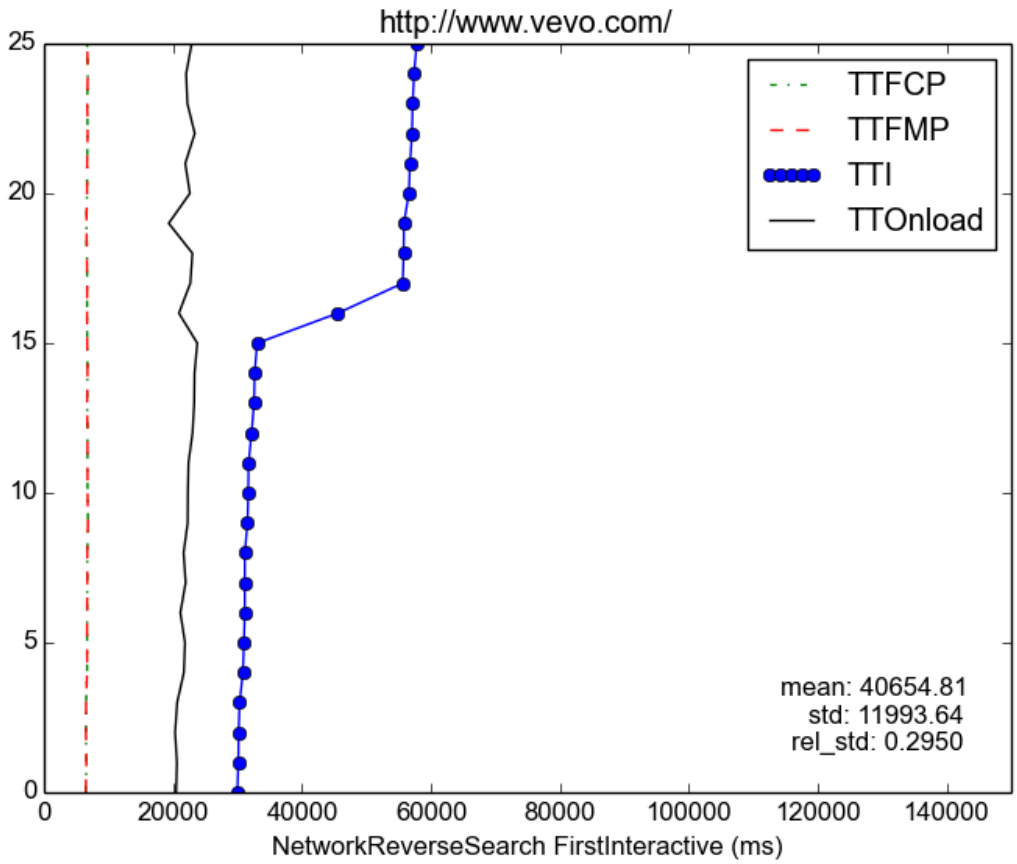
We should ideally fix TTI so that we fix the variance in case 1 - although it is possible that if we strive for a moment-in-timeline metric like firstInteractive there will always be these edge cases no matter how we define the metric. The right approach may be to combine TTI and EQT in different phases into one unified metric which will have less variance than its components, and then monitor that unified metric.

## Data

You can look at all the graphs in one place here. Some summary statistic per site can be found here. Data for each individual run along with a link to the traces (access to individual traces are Googler only) can be found here.

## Digging into cases with high variance

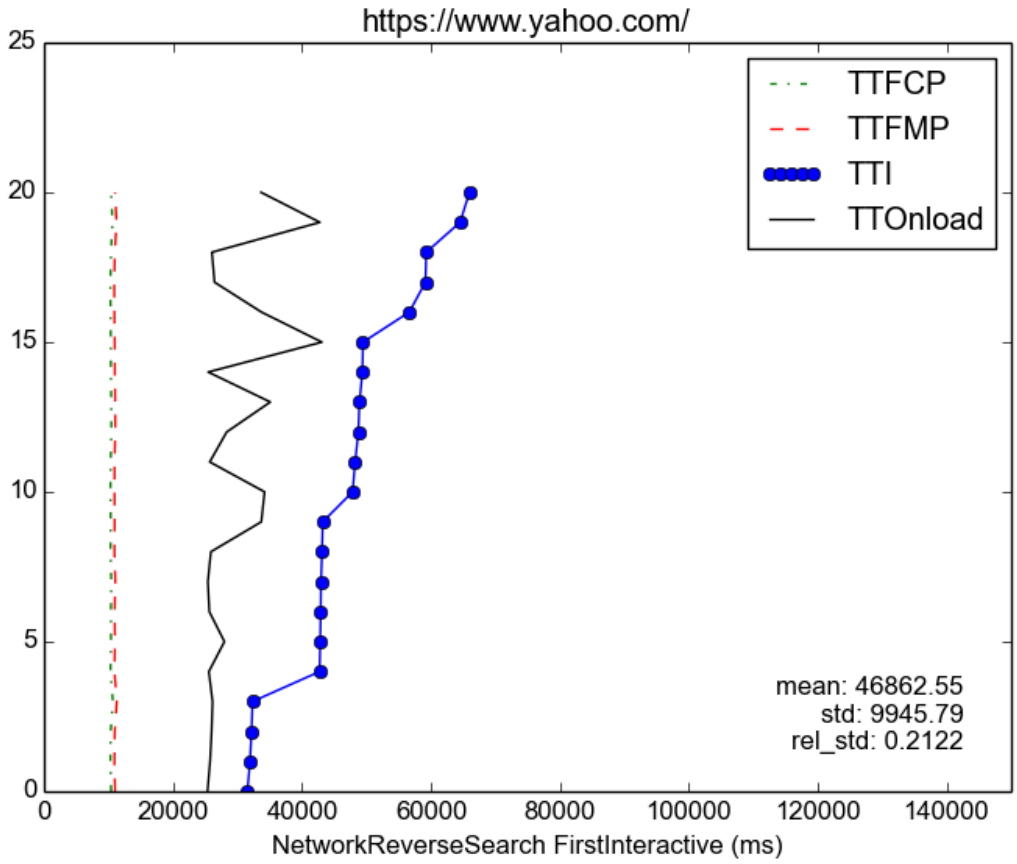
### Case 1: Vevo.com - clear bimodality




~15 runs got a TTI value around 30s, but the rest got closer to 60s. This all comes from a ~60ms MajorGC task that fires pretty consistently around the 60s mark. There is quite a bit of network traffic (js for analytics / js talking to json api about non-critical data / non-critical images) after the page is fully loaded, and sometimes these requests prevent us from getting 5 seconds of network quietness we need to stop our search for TTI. If network quietness happens after the GC task, our reverse search identifies the GC task as a long task and fires firstInteractive at the end of it.

- Example trace with firstInteractive after MajorGC. Another one.
- Example trace with firstInteractive before MajorGC

**Case 2: yahoo.com - values are all over the place**




The TTI values can be segmented into four regions:

- 31.5s - 32.5s | Example trace
- 42.7s - 43.3s | Example trace
- 47.8s - 49.5s | Example trace
- 56.6s - 66s | Example trace

Here is a comparison of these four traces:

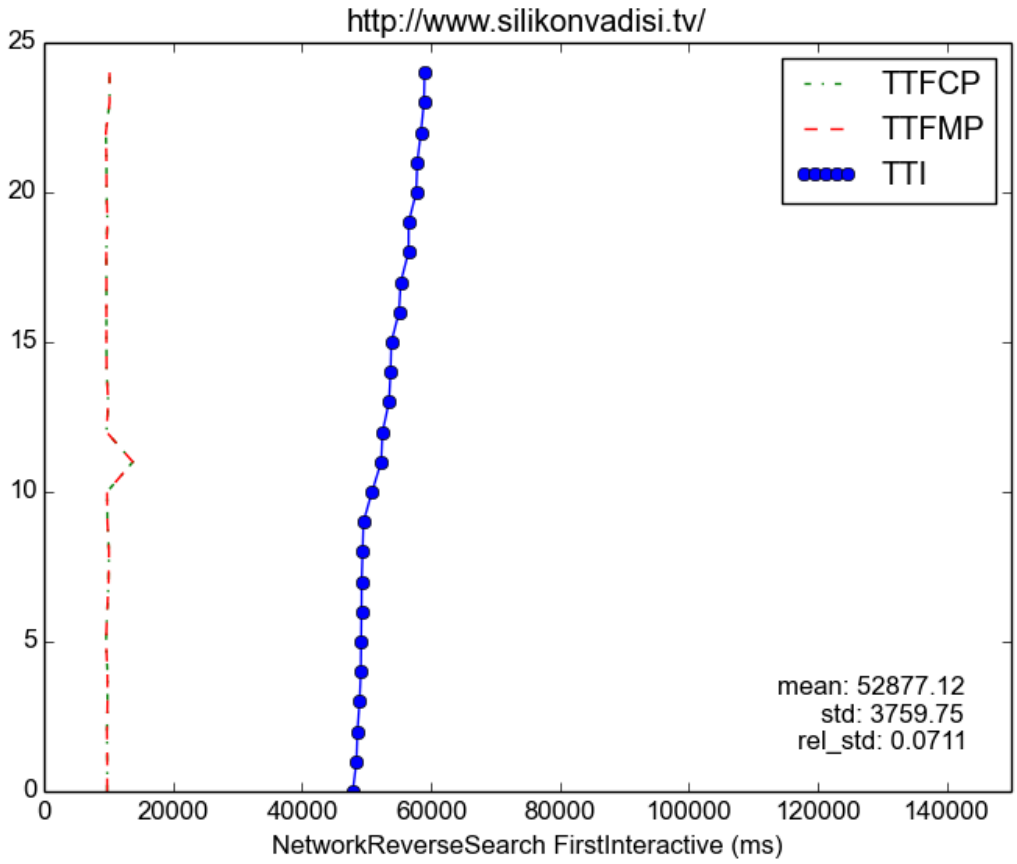
See bigger image here. Interactive version here, but beware - this can take minutes to load.

#1 and #2 are fairly similar. In example #1, we see a large group of long tasks from 18-38s, and another group of long tasks around 45-50s. In example #2, the first group of long tasks finish at 36s, but the second group starts at 41s, and we end up not having the 5 seconds of main thread and network quietness that is needed to first firstInteractive early. Here is a side by side comparison of the long tasks in the two situations.

#3, and especially #4 is more complicated - the trace from those runs indicate the page executed differently than the first two runs. The cause is unclear: it is possible this is because the page loading different sets of ads - maybe it loaded a video ad this time? See here for the list of long tasks.

Our TTI metric *should* behave differently in this case. If you want to use TTI in a CI environment we should probably either always use pre-recorded network traffic, or take the average of multiple runs.

**Case 3: http://www.silikonvadisi.tv/ - large number of long tasks**

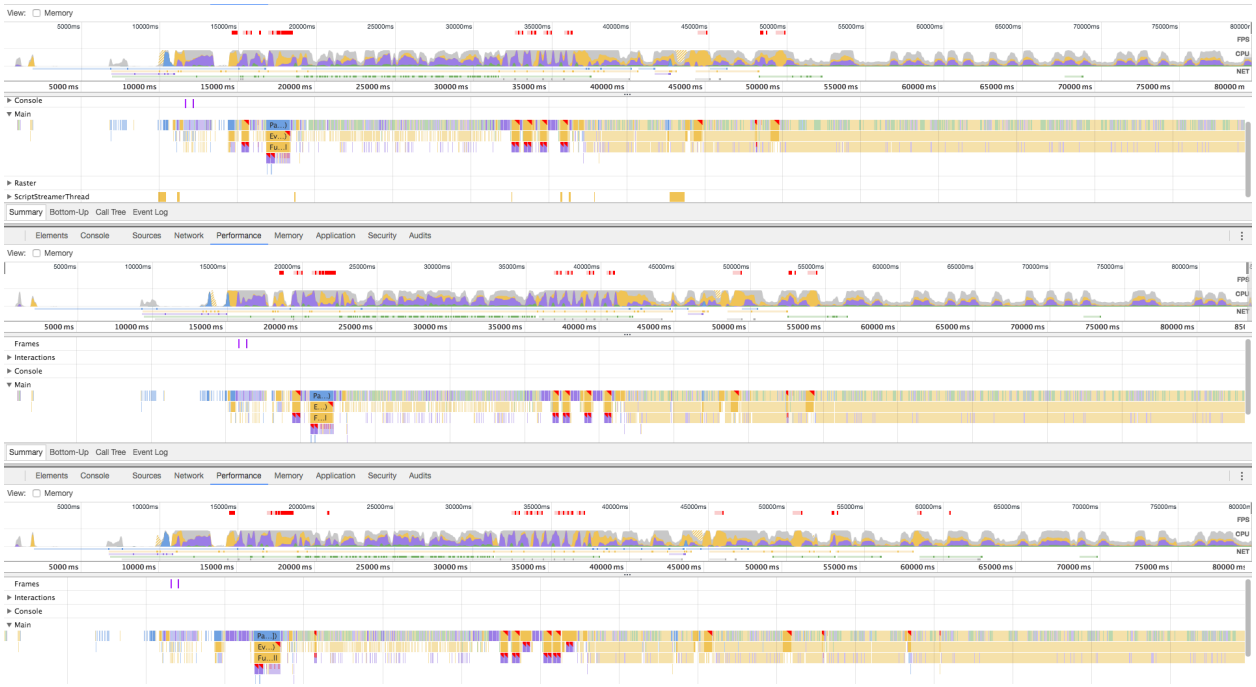



TTI value ranges from 47s to 60s, while FMP and FCP remains relatively stable. We can look at three traces:

- Trace with TTI of 47.9 seconds
- Trace with TTI of 52.2 seconds
- Trace with TTI of 58.5 seconds

Json versions can be found here if you want to load it in devtools.

Here is a side by side comparison of the three traces:




[Bigger version here.]

Clearly there is a *lot* of things going on in this site. [See here for a list of long tasks.] All three of these traces have **more than 100** tasks longer than 50ms. A large number of these tasks are ThreadProxy::BeginMainFrame, which groups together several update-layer-tree and paint tasks, and it is the update-layer-tree tasks that are routinely more than 50ms long. The list of long tasks end with a large ~500ms task, and firstInteractive usually fires right after or very close to the end of that task. firstInteractive here does not really jump around among several points of the timeline - the most probable cause of the wide 13s band of values is the accumulation of variability in execution time of all the long tasks.

---

*Chrome speed metrics team can be reached at progressive-web-metrics@chromium.org.*