NutShell: Scalable Whittled Proxy Execution for Low-Latency Web over Cellular Networks
A review.

Kevin Dackow

July 2018

1 Summary

NutShell addresses the page load bottleneck that cellular network latency causes on mobile phones by minimizing unecessary processing committed by redundant execution on proxy servers. Redundant execution is a strategy employed by proxy servers to reduce latency by having the proxy execute the code for a page and then push the objects needed for page load to the client. The goal is to send data to the client before the client requests it, to save valuable download time. This runs into the problem of high loads on the proxy servers, especially as the number of clients scales. To tackle this issue, NutShell points out that executing the entire page on the proxy is unecessary, because rendering does not happen server-side. Thus, NutShell proposes that only scripts related to fetching URLs should be executed on the proxy, and that the rest of the code can be ignored.

To do this, NutShell proposes a technique caleld Whittling for cutting out all JS that is not needed for identifying fetched objects. This amounts to backward slicing the code related to the URLs to be fetched. To do this Whittling, NutShell relies on a learning model that takes this form:

1. Process the full page and create a list of all fetched URLs

2. Rank all JS functions from highest CPU usage to lowest.

3. Remove as many functions as possible so that the set of fetched URLs remains the same as the full page rendering (note: remove a function = make the declaration empty)

After removing all unecessary JS functions, the code is then executed on the server-side (with errors ignored) and the list of URLs is sent to the client.

It is important to note that NutShell reluies on calculating the whittled code one time per site for every 3 hours, as it is clear that whittling for every client request would be slower than traditional proxy approaches.

2 Analysis