Narrowing Down the Debugging Space of Slow Search Response Time

Dapeng Liu†, Youjian Zhao†, Dan Pei†* , Chengbin Quan†
Qingqian Tao‡, Pei Wang‡, Xiyang Chen‡, Dai Tan‡, Xiaowei Jing§, Mei Feng§
†Tsinghua University ‡Baidu §PetroChina
†Tsinghua National Laboratory for Information Science and Technology (TNList)

***Abstract*—When using search engines, users often care about search response time (SRT) in addition to result accuracy. It is thus the operators' responsibility to closely monitor and improving SRT. The first critical step of improving SRT is to pinpoint the root causes of slow SRT. However, this task is very challenging because SRT can be impacted by many factors, e.g., networks, data centers, browsers, and the page content.**

In this paper, we propose FOCUS, a systematic framework to narrow down the debugging space of slow SRT by identifying the bottleneck of slow SRT regarding various factors. The bottleneck provides operators more specific direction for further investigation. We deployed FOCUS in a global top search engine. Based on the output of FOCUS, operators successfully identified four potential causes which would not have been easy to find without FOCUS. Our what-if simulation analysis shows that, the proposed solutions, focusing on these bottlenecks, can improve SRT significantly, and they are more effective than some ad hoc solutions.

I. Introduction

Search engine is no doubt one of the most prevalent applications of the Internet. Billions of queries are launched from all over the world everyday, and then handled by search engines such as Google, Baidu, Yahoo, Yandax, and Bing [1]. Operating such a giant search system is very challenging, and operators have to work very hard to satisfy dozens of KPIs (key performance indicators). Among them, search response time (SRT) is one of the biggest concerns for search providers [2]. SRT refers to the user perceived waiting time between when a query is submitted and the time when the result page is fully rendered. As such, SRT has a measurable impact on users' experience as well as providers' profit. [3], [4] found that less than half a second increase of SRT can lead to 0.6% fewer searches and 1.2% drop in revenue. As a result, operators are responsible for monitoring and improving SRT, especially the slow SRT, so that it can satisfy the growing requirement. For example, the search engine we studied requires the 80^{th} percentile of SRT less than 1 second.

Recently, many acceleration solutions have been proposed [5], [6], [7], [8], [9], [10], [11] . They aim to fix particular problems in the fourth step. Yet, a key missing part before applying a solution is to identify the bottleneck of slow SRT. In particular, we want to answer the following two questions: Under which conditions queries are slow? Which components of SRT is slow? These two questions can help operators debug slow SRT.

In this paper, we propose a novel systematic framework, called FOCUS, to systematically answer the above two questions. FOCUS intends to automatically identify the *bottleneck conditions*, and the *bottleneck SRT components*. The results, outputted by FOCUS, provide operators specific investigation directions and enable them to further identify the root causes. For example, if we find that many queries triggering ad are responded slowly, and their DOM (document object model) load time is long, operators should investigate whether the modual regarding ad is inefficient, or contains some bugs.

* Dan Pei is the corresponding author.

The task of FOCUS in practice is challenging due to the following aspects. First, SRT can be affected by many factors such as servers, networks, browsers, and users' devices. Second, since these factors inherently overlap each other, it is difficult to identify which factor is responsible for the slow SRT. For example, a condition that Chrome runs on a less powerful device. Third, the output should be specific and straightforward for operators. For example, the output of traditional clustering methods like k-means do not have clear boundaries, thus unintuitive.

To tackle these challenges, we first develop a multi-dimensional hierarchy clustering to provide clear and meaningful boundaries of the bottlenecks. This ensures that the clusters we find out are specific for operators. Then we leverage a technique called hierarchical heavy hitter (HHH) [12], that has been commonly used to locate iceberg in network traffic. This technique can help identify in the multi-dimensional hierarchy which clusters are the real bottlenecks and which ones are redundant. Once bottleneck clusters have been found, we design a method based on Occam's razor to further determine which SRT components can best explain the slow SRT.

We deploy FOCUS in one of the global top search engines. Based on the bottlenecks identified by FOCUS, operators successfully locate four causes of slow SRT and propose solutions. Our what-if simulation further demonstrates that, the solutions focusing on the bottlenecks by FOCUS are much more effective than ad hoc solutions in improving SRT. Some of these ad hoc solutions were actually being considered by the operators for deployment in the studied search engine before using FOCUS. These results highlight the value of FOCUS in the field of debugging slow SRT.

The remainder of the paper is organized as follows. Section II provides the basic background of SRT and our problems. Section III describes the details of FOCUS. Section IV shows the results of FOCUS over real data and the simulation. Section V reviews the related work, and Section VI concludes the paper.

II. Background and Problem

A. About SRT and Requirement

To better understand SRT, we first introduce the events happened after submitting a query. Instead of discussing the details, we here only provide a simplified view to build high level intuitions. Fig. 1 shows five steps. (1) When query is submitted (if the result is not cached by the client), the host name of the search engine will be resolved by the the provider's DNS. The DNS responds the IP address of a close data center. (2) Then the browser sends a query to the search data center. The data center conducts a series of complex processes, such as results ranking, ad strategies, and page constructing, before sending the result page to the browser. (3) The browser starts parsing the page and loading DOM. (4) Embedded images of the page are

978-1-4673-8590-9/15/$31.00 ©2015 IEEE