Computer Networks 91 (2015) 298–315

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

A hybrid local and distributed sketching design for accurate and scalable heavy key detection in network data streams☆

Qun Huang, Patrick P.C. Lee*

Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong

ARTICLE INFO

Article history:
Received 4 November 2014
Revised 17 July 2015
Accepted 27 August 2015
Available online 8 September 2015

Keywords:
Sketches
Heavy hitter/changer detection
Streaming architectures
Network monitoring

ABSTRACT

Real-time characterization of network traffic anomalies, such as heavy hitters and heavy changers, is critical for the robustness of operational networks, but its accuracy and scalability are challenged by the ever-increasing volume and diversity of network traffic. We address this problem by leveraging parallelization. We propose LD-Sketch, a data structure designed for accurate and scalable traffic anomaly detection using distributed architectures. LD-Sketch combines the classical counter-based and sketch-based techniques, and performs detection in two phases: local detection, which guarantees zero false negatives, and distributed detection, which reduces false positives by aggregating multiple detection results. We derive the error bounds and the space and time complexity for LD-Sketch. We further analyze the impact of ordering of data items on the memory usage and accuracy of LD-Sketch. We compare LD-Sketch with state-of-the-art sketch-based techniques by conducting experiments on traffic traces from a real-life 3G cellular data network. Our results demonstrate the accuracy and scalability of LD-Sketch over prior approaches.

1. Introduction

Characterizing traffic anomalies is important for administrators to maintain the robustness of a network. There are two types of anomalies that are of particular interest: flows with persistently large data volume (known as "heavy hitters") and flows with abrupt changes of data volume (known as "heavy changers"), since they may imply the existence of denial-of-service attacks, component failures, or service level agreement violation. Traffic anomalies are detrimental to network robustness and must be detected and suppressed in real-time.

However, today's IP networks continuously grow in size and complexity. Thus, characterizing traffic anomalies in real-time becomes more challenging, particularly in two aspects:

- *Enormous key space.* The complexity of anomaly detection is overwhelmed by the number of keys being monitored. For example, 5-tuple network flows are defined by 104-bit keys (i.e., source/destination IP addresses, source/destination ports, and protocols). Keeping track of 2^{104} flow keys can imply huge memory usage.
- *Line-rate packet processing.* Packet processing must keep pace with the increasing line rate to meet the real-time requirement. Conventional single-processor platforms no longer provide enough computational power to achieve this goal. To improve scalability, anomaly detection needs to be performed on *distributed* packet streams in parallel. However, providing both accuracy and scalability guarantees becomes challenging when aggregating the detection results from multiple sources.

☆ **Note:** An earlier conference version of the paper appeared in IEEE INFOCOM 2014 [15]. We extend our proposed design with the consideration of the ordering of data items (Section 5). We also correct the flaws of the lemmas and theorems in our conference version.

* Corresponding author. Tel.: +852 39434260.
E-mail addresses: qhuang@cse.cuhk.edu.hk (Q. Huang), pclee@cse.cuhk.edu.hk (P.P.C. Lee).

http://dx.doi.org/10.1016/j.comnet.2015.08.025
1389-1286/

Anomaly detection has been extensively studied in the context of data streaming. To deal with the enormous key space, counter-based techniques (e.g., [12,19–21]) and sketch-based techniques (e.g., [4,5,7–11,13,16,17,23]) propose space-efficient data structures for anomaly detection and derive error bounds. Counter-based techniques use an associative array to monitor frequent items and are designed for heavy hitter detection; sketch-based techniques project data items into a subset of buckets in a summary called *sketch* and are designed for both heavy hitter and heavy changer detections. Although theoretically sound, such techniques are mainly studied and evaluated in the single-processor paradigm. With the emergence of distributed streaming architectures (e.g., Flume [2], S4 [22], and Storm [25]), an open issue is to seamlessly parallelize such techniques to achieve accurate and scalable anomaly detection.

In this paper, we study the traffic anomaly detection problem from both theoretical and implementation perspectives. We propose *LD-Sketch*, a novel sketching design that combines the counter-based and sketch-based techniques to accurately and scalably detect heavy hitters and heavy changers using distributed architectures. Its main idea is to augment a sketch in which each bucket keeps track of anomaly candidates in an associative array, similar to counter-based techniques. We enhance our counter-based technique to allow the associative array to be dynamically expandable based on the current number of anomaly candidates associated with the bucket. LD-Sketch performs detection in two phases: (i) *local detection*, which guarantees no false negatives and identify the anomaly candidates (including true anomalies and false positives) in a single compute node called *worker*, and (ii) *distributed detection*, which reduces false positives (with a slight increase in the false negative rate) by combining detection results from multiple workers. Thus, not only do we exploit streaming architectures to improve scalability, but we also use their distributed nature to improve the detection accuracy.

In summary, we make the following contributions:

- We design LD-Sketch, which enables accurate and scalable detection of heavy hitters and heavy changers and is seamlessly deployable in distributed architectures. We derive the error bounds, space complexity, and time complexity when LD-Sketch is used in both local detection (i.e., using a single worker) and distributed detection (i.e., using multiple workers).
- We analyze how the ordering of data items influences the memory usage and accuracy of LD-Sketch. By leveraging the ordering property of data items, we propose two enhancement heuristics that respectively reduce the memory usage and the false positive rate of LD-Sketch.
- We implement and compare LD-Sketch with state-of-the-art sketch-based techniques by conducting trace-driven experiments using traces from a real-life 3G cellular network. We show that LD-Sketch achieves higher accuracy than other approaches, and improves accuracy and scalability using multiple workers in a distributed setting.

The rest of the paper proceeds as follows. Section 2 formulates the heavy hitter/changer detection problem. Sections 3 and 4 describe and analyze the local and distributed detection procedures of LD-Sketch, respectively. Section 5 studies the impact of ordering of data items and proposes two enhancement heuristics for LD-Sketch. Section 6 presents our trace-driven evaluation results. Section 7 reviews related work, and finally Section 8 concludes the paper. We also present our proofs of lemmas in Appendix.

Table 1
Major notation used in the paper.

Notation	Meaning
Defined in Section 2	
p	Number of remote sites
q	Number of workers
$[n]$	Key domain
(x, v_x)	Data item with key x and value v_x
$S(x)$	True sum for key x in an epoch
$D(x)$	True difference for key x between two adjacent epochs
U	Total sum of the values of all keys in an epoch
ϕ	Heavy key threshold
H	Maximum possible number of heavy keys
r	Number of rows in a sketch
w	Number of buckets in one row in a sketch
(i, j)	The jth bucket in row i, where $1 \le i \le r$ and $1 \le j \le w$
f_i	Hash function $\{0, 1, \ldots, n-1\} \to \{1, 2, \cdots, w\}$ for row i
Defined in Section 3	
$V_{i,j}$	Counter value of bucket (i, j) in the sketch
$A_{i,j}$	Associative array in bucket (i, j) ($A_{i,j}[x]$ denotes the counter value of key x)
$l_{i,j}$	Maximum length of array $A_{i,j}$
$e_{i,j}$	Maximum error for keys in bucket (i, j)
T	Expansion parameter
k	Current expansion number (starting from zero)
$S^{\text{low}}_{i,j}(x)$	Lower estimated sum for key x in bucket (i, j)
$S^{\text{up}}_{i,j}(x)$	Upper estimated sum for key x in bucket (i, j)
$D_{i,j}(x)$	Estimated difference for key x in bucket (i, j)
ϵ	Approximation parameter in local detection
δ	Upper bound of error probability
Defined in Section 4	
γ	Approximation parameter in distributed detection
d	Number of workers to which a key is distributed
Defined in Section 5	
$\hat{k}_{i,j}$	Controlled expansion number
$t_{i,j}$	Number of distinct keys in bucket (i, j)
$t^{\ge}_{i,j}$	Number of keys satisfying $S(x) \ge T$ in bucket (i, j)
$V^{\ge}_{i,j}$	Total sum of the values of all keys satisfying $S(x) \ge T$ in bucket (i, j)
$V^{<}_{i,j}$	Total sum of the values of all keys satisfying $S(x) < T$ in bucket (i, j)
$k^{\ge}_{i,j}$	Integer such that $k^{\ge}_{i,j}T \le V^{\ge}_{i,j} < (k^{\ge}_{i,j}+1)T$
$k^{<}_{i,j}$	Integer such that $k^{<}_{i,j}T \le V^{<}_{i,j} < (k^{<}_{i,j}+1)T$

2. Problem formulation

We formulate the problem of heavy hitter/changer detection (which we collectively call *heavy key detection*). Table 1 summarizes the major notation in this paper.

2.1. Distributed heavy key detection

We first describe the distributed architecture, as shown in Fig. 1, on which our heavy key detection problem is formulated. Our architecture is based on that of [6]. Specifically, we consider an architecture with $p \ge 1$ *remote sites* and $q \ge 1$ *workers*. A remote site is the source of a data stream, while a worker performs heavy key detection based on the streams from multiple remote sites. The architecture has a bipartite