计算机研究与发展
Journal of Computer Research and Development
DOI:10.7544/issn1000-1239.2020.20190287
57(5): 1046-1056, 2020

分布式监测系统中的重复元素检测机制

陆 乐[1] 孙玉娥[2,3] 黄 河[1,3] 汪润枝[1] 曹 振[1]
[1](苏州大学计算机科学与技术学院 江苏苏州 215131)
[2](苏州大学轨道交通学院 江苏苏州 215137)
[3](中国科学技术大学苏州研究院 江苏苏州 215123)
(20175227062@stu.suda.edu.cn)

Detection of Persistent Elements in Distributed Monitoring System

Lu Le[1], Sun Yu'e[2,3], Huang He[1,3], Wang Runzhi[1], and Cao Zhen[1]
[1](*School of Computer Science and Technology, Soochow University, Suzhou, Jiangsu* 215131)
[2](*School of Rail Transportation, Soochow University, Suzhou, Jiangsu* 215137)
[3](*Suzhou Institute for Advanced Study, University of Science and Technology of China, Suzhou, Jiangsu* 215123)

Abstract The detection of persistent elements has many important applications in the fields of detecting intrusions in a distributed system, finding common interests, measuring the traffic, etc. Most of the existing state-of-the-art studies of detecting persistent elements have some problems such as false or missing report, high communication cost and great limitation so that they can hardly satisfy the requirement of some distributed applications. To solve these problems, the paper proposes a scheme to detect persistent elements in distributed monitoring system with the goal of minimizing the total communication cost during the whole detection process. First, the scheme filters out most of the irrelevant elements to reduce the overall communication overhead through multiple rounds of compressed data transferring between all monitors and the central coordinator. Then, we ensure that each round of the filtering is necessary and can achieve the best performance by adjusting parameters of filtering according to the theoretical analysis and derivation. With the technology of extended Bloom filter and the persistent spread estimation function, the scheme can work well no matter in the balanced environment or in the unbalanced environment. Finally, we perform extensive simulations to study the performance of the proposed mechanism, and the simulation results show the effectiveness of our scheme.

Key words distributed system; persistent element; Bloom filter; communication cost minimization; mechanism design

摘 要 重复元素检测在分布式入侵检测、公众兴趣发掘以及交通状况估计等领域有着重要的应用.现有的检测模型存在误报漏报、通信开销大和局限性大等问题,难以满足分布式应用场景的需要.针对这些问题,以最小化通信开销为目标,设计了一个适用于分布式监测系统的重复元素检测机制.首先,机制主要通过监测器与协调器间多轮次的压缩数据传输筛去了大量无关元素,从而降低了整体通信开销;接

收稿日期:2019-05-16;**修回日期**:2019-11-29
基金项目:国家自然科学基金面上项目(61672369,61873177,61572342)
This work was supported by the General Program of the National Natural Science Foundation of China (61672369, 61873177, 61572342).
通信作者:孙玉娥(sunye12@suda.edu.cn)

着,借助理论推导调整参数保证每一轮筛选的必要性及效果的最优化,并结合了可扩展布隆过滤器和持续流量估计等技术,使机制在面对不同的元素分布状况时,都可以取得良好效果;最后,通过仿真实验结果验证了所设计机制的有效性.

关键词　分布式系统;重复元素;布隆过滤器;通信开销最小化;机制设计

中图法分类号　TP393

近年来,随着分布式服务架构的流行,人们越来越关注分布式系统中各节点的数据整合问题.如何通过尽可能少的通信开销整合各分布式节点存储的数据,并从中提取出有用的信息来帮助系统做出更好的决策,已经成为人们研究的重点之一.给定一个阈值,当一个元素被不少于该阈值个节点监测到时,我们就称该元素为重复元素.本文将研究分布式系统中的重复元素检测问题,即分布式系统中同时被不少于给定阈值个节点监测到的重复元素集合,属于节点数据整合问题的一种.对重复元素的有效检测在实际生产中有很多重要的应用,举出一些应用实例:

1) 在一个包含了大量可以独立检测恶意入侵行为节点的分布式入侵监测系统[1-2]中,每个节点只检测自己监控的区域是否遭到了入侵并记录入侵的相关信息,如入侵类型、地址等能区分不同入侵的数据.通过网络,这些节点将入侵记录上传到中央节点,由中央节点找出并重点处理其中威胁度较高的大规模恶意入侵记录,即被大多数节点检测到的入侵数据.

2) 如谷歌一类的在线搜索公司在世界各地都有自己的服务器,假设它们会记录下各自地区近期内搜索频次较高的热门关键词并将其传给中央服务器,那么中央服务器就能整合这些搜索数据,找出其中大部分地区的人的共同兴趣或者关注点.比如在谷歌流感趋势预测[3]中,谷歌通过收集和整合各个地区人们跟流感相关的搜索关键词数据,可以近乎实时地预测流感是否爆发以及流感爆发的区域.

目前的相关研究主要集中在分布式系统中对频繁元素[4-8]和全局元素[9-14]的检测上.其中频繁元素即为在整个系统中出现频率最高的 s 个元素,在这一类问题的模型中,同一元素可以被一个节点记录多次.文献[4]针对频繁元素检测问题提出了一个多层次检测模型,将节点划分为多个层次,通过中间节点的数据整合减小最终传输给中央协调器的数据量,但该方案并不能有效降低整个过程中消耗的通信代价;文献[5]先让各节点找出本节点中出现频次最高的 s 个元素,然后将其发送给协调器以计算出一个更低的频次阈值并发回给各节点,然后节点再将出现频次超过该阈值的所有元素发给协调器,由其找出其中的频繁元素.然而,上述研究并不适应于重复元素检测.这是因为重复元素检测要计算每个元素被多少个不同节点监测到,而与该元素在每个节点上出现的次数无关.与本文研究问题类似的是全局元素检测,即找到被所有节点监测到的元素,是重复元素中的一个特例.文献[9]中提出一种基于任意 2 个节点记录的元素集合差异较小的假设的全局元素检测算法,但这一假设与实际不符,在节点差异较大的情况下其通信开销甚至超过了直接传输原数据的直观算法;文献[10]中提出的 CFSP(combinable filter solution with progressive filtering)算法通过布隆过滤器[15]对节点数据进行压缩,并在协调器端将压缩数据通过逐位相与的方式整合后发回各节点以筛去节点中的无关元素,最终经过多轮筛选在给定的误差范围内找出系统中的全局元素集合.CFSP 算法可以有效降低检测过程中的通信开销,但其在参数设置过程中并没有从全局最优的角度进行参数优化,所以在理论上并没有取得最佳效果.此外,全局元素本身也存在着很大的局限性.以分布式入侵监测系统为例,入侵者如果有意识地漏过 1～2 个节点不去攻击,就不会被全局元素检测算法检测到.因此本文在 CFSP 算法的基础上,提出了一个更为普适的问题模型,并给出了一个检测重复元素的有效解决方案。此外,本文还通过概率推导从全局最优的角度对参数进行了优化,并且增设了对筛选停止条件的判断,从而保证了每一轮筛选的收益最大化。

与本文最为相关的研究是文献[16]中提出的利用 Cuckoo Filter 和 Raptor Code 来检测重复元素的 DISPERSE(probabilistic distributed persistent item identification algorithm)算法,但误报和漏报情况的存在使其难以适用于某些严苛的应用场景.

在重复元素的检测过程中存在 4 个难点:

1) 大规模的分布式系统会产生巨量的数据,这

些数据的传输会给协调器和整个系统造成极大的压力,因此需要尽可能降低探测过程中的通信开销,避免原始数据的直接传输.

2) 系统中不同节点记录的数据量可能大致相仿,也可能差异巨大,机制针对不同的元素分布状况,都要能达到预期的效果.

3) 为了便于统筹全局,做出决策,协调器需要掌握所有重复元素的信息,而这些数据都是分散存储在各个节点中的.

4) 在某些应用场景下,重复元素的误报或漏报可能造成巨大的损失,为了适用于类似场景,机制要完整、无误报地检测出所有重复元素.

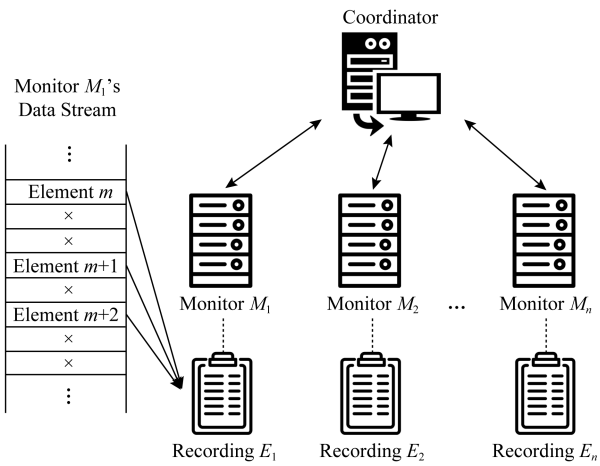
针对以上 4 个难点,本文以降低通信开销为目标,设计了一个适用于大规模分布式监测系统的重复元素检测机制,保证协调器可以完整、高效、无误报地完成重复元素的检测.

1 系统建模

本节给出了分布式监测系统的模型,并对所要解决的问题进行了具体阐述.

1.1 系统模型

如图 1 所示,所研究的分布式监测系统部署在网络中,主要由 1 个中央协调器和 n 个监测器组成.每个监测器独立地监测网络数据流中的元素,这里的元素可以根据实际的应用需求定义.例如,在扫描攻击检测应用中,每台监测器是一个服务器,而一个元素则是一个访问该服务器的源地址.由于本文只研究如何检测网络中的重复元素,并不需要考虑每个元素在同一监测器中出现的次数,所以假设同一监测器所保存的元素集合并不存在相同元素(即已经完成去重操作).



The symbol "×" in monitor M_i's data stream represents the data which is not recorded by the monitor M_i.

Fig. 1 The distributed monitoring system model

图 1 分布式监测系统的系统模型

当需要进行重复元素检测时,每个监测器 M_i 首先开辟一块大小为 l_i 的内存空间,用于存储所监测到的元素集合 E_i.由于元素的数量可能非常庞大,直接发送原始数据会带来巨大的通信负担.为了节省传输带宽,本研究采用布隆过滤器存储元素,而每个监测器上布隆过滤器的大小 l_i 由所监测到的元素数量决定.然后,监测器将编码后的布隆过滤器发送给协调器.协调器在收到所有监测器发来的数据后会对其进行整合,然后通过与监测器之间的多轮通信、协作计算完成重复元素监测工作.

1.2 问题描述及设计目标

实际应用中,根据应用场景的不同,元素的定义可以非常宽泛.它可以是 1 个搜索频率极高的热门关键字、1 条被蜜罐捕获的入侵记录或是 1 个被检测系统判定为攻击源的 IP 地址.在本文中,任意 2 条元素,只要内容相同,就会被认定是同一条元素.因此,1 条元素可以出现在多个不同监测器的记录列表中.下面我们给出 t-重复元素的定义,这也是本文研究的主题.

定义 1. t-重复元素.在一个检测周期中,如果某条元素被 n 个监测器中的至少 t 个记录下来,那么称该条元素为 t-重复元素.

本文研究的目标是通过监测器与协调器间的 1 轮或多轮通信,筛去各监测器列表上的非 t-重复元素,使协调器能用尽可能少的通信资源,找到一个包含了本周期内系统中出现的所有 t-重复元素的集

合,此处记为 P_t.参数 t 作为界定 t-重复元素的阈值,其数值的改变会很大程度上影响 P_t 的内容.在 t 取不同数值的情况下,P_t 可能包含大量甚至全部的元素,也可能仅仅是某个集合 E_i 的一个子集.

在寻找 P_t 的过程中,本文提出的机制需满足 3 点设计目标.

1) 完整性.在一些严格的应用场景下,系统必须保证协调器找出的解是没有遗漏的,即集合 P_t 要包含所有 t-重复元素.

2) 无误报.最终解不能存在任何误报的情况,任何一个非 t-重复元素都不应属于 P_t.

3) 筛选的必要性.机制中进行的每一轮筛选操作都应该是必要的,即相对直接发送剩余元素原始数据的方案而言,本轮的筛选能够确实有效地降低整体的通信开销.

基于以上考虑,本文以最小化中央协调器与监测器之间的数据传输总量为优化目标,提出了一个高效的 t-重复元素检测机制,在保证监测准确率的基础上,尽可能地降低对传输带宽的占用.

2　t-重复元素检测机制

本节给出了 t-重复元素检测机制的设计细节,主要包括检测流程以及前 2 轮和后续轮次筛选过程中的参数优化,并对机制的性能进行了分析.

2.1　检测流程

对于 t-重复元素检测问题而言,一种简单直观的做法就是令各监测器直接将元素列表全部发送给协调器,协调器经过比对后就能准确无误地找出所有 t-重复元素.这种方法思路简单、易于操作,但会产生巨量的通信消耗,在大规模分布式监测系统中尤其如此.而要让协调器检测到存储在各个监测器端的所有 t-重复元素,这些 t-重复元素的直接传输又是必要的.因此,为了降低整体的通信成本,本机制的基本思路是通过监测器与协调器之间多轮的压缩数据交互,筛去监测器端元素列表中绝大部分的非 t-重复元素,从而减小需要直接传输的元素的数量.

当一个监测周期结束时,系统即开始本周期的 t-重复元素检测.由于布隆过滤器具有结构简单、空间复杂度低以及无漏报等优点,本文将其作为监测器压缩元素列表的工具.在用布隆过滤器对元素列表进行编码之前,监测器需要先用计算得到的布隆过滤器的参数,即位数组的大小 l 和 Hash 函数数量 k,对其进行初始化.初始化完成后,各监测器首先将列表中的所有元素在对应的布隆过滤器上置位.

具体的置位流程为:对于每个元素,布隆过滤器通过给定的 k 个 Hash 函数将其随机映射到长为 l 的位数组上的 k 个位置,并将这些位的值置为 1.置位完成后,监测器将各自的位数组发送给协调器,由协调器将接收到的所有位数组通过逐位累加的方法进行整合,把累加后数值大于等于 t 的位置为 1,其余位则置 0,从而得到一个整合了所有元素数据的结果位数组.此后这个结果位数组会被协调器发回给所有监测器,每个监测器对照着它用原来的 k 个 Hash 函数将自身列表中的元素一一映射到对应的 k 位,若这 k 位全部为 1,那么这条元素可能是 t-重复元素,监测器依旧将其保留.若这 k 位不全为 1,基于布隆过滤器无漏报的特性,该元素一定是非 t-重复元素,监测器会将其从列表中移除.当所有元素都在结果位数组上验证过一遍后,这轮筛选便到此结束.

系统重复上述步骤,进行下一轮的筛选,直到监测器接收到协调器的终止命令为止.筛选结束后,监测器将列表中剩余的所有元素直接发送给协调器,协调器通过对比找出所有 t-重复元素.

在筛选阶段,使用布隆过滤器进行数据的编码压缩在判别时总会有一定的误报率存在,所以即使经过多轮次的筛选,机制也无法保证所有的非 t-重复元素都已经被监测器筛去.但无论监测器最终传输的元素中是否包含非 t-重复元素,机制都能确保协调器可以准确地找到 P_t.因此筛选阶段允许误报的存在,且本文不关心误报率的大小,下一轮筛选对整体误报率的影响并不能直接决定筛选是否继续进行.后续筛选的必要性只取决于其是否能给系统整体带来收益,即通信开销的降低,而这一点将交由协调器判断.为了让协调器能够通过逐位累加的方法整合布隆过滤器,所有监测器对应的布隆过滤器理论上应该保持一样的大小 l,并使用相同的 k 个 Hash 函数进行置位.

记第 q 轮筛选中监测器 M_i 对应的布隆过滤器为 $B_{i,q}$,其对应的位数组长度为 $l_{i,q}$,统一的 Hash 函数数量为 k_q.紧接着将介绍协调器怎样判断筛选何时停止以及如何给各布隆过滤器分配合适的参数,因为不同阶段元素集合的情况有所差异,本机制为前 2 轮筛选和后续筛选分别设计了不同的参数优化方案.

2.2 首轮筛选机制

第 1 轮筛选中,考虑到各监测器所记录的元素数目可能相差极大的情况,为所有布隆过滤器统一设置一个较大的 l 可能会在空间上造成极大的浪费,而较小的 l 又无法满足所有监测器的筛选需要,使得筛选无法达到预期的效果.因此本机制引入了文献[10]中提出的可扩展布隆过滤器技术,根据每个监测器所记录的元素数量,对应的布隆过滤器会被单独分配一个位数组长度 $l_{i,1}$,这是由协调器综合考虑了全局情况后通过计算得到的最优参数.但是,在实际初始化布隆过滤器的过程中,位数组长度会被设置为 $2^{\lfloor \mathrm{lb}\, l_{i,1} \rfloor}$b. 当所有布隆过滤器完成编码并被发送给协调器后,因为所有位数组长度都是 2 的幂次方,协调器可以将每一个过滤器通过若干次自我复制使其长度扩展到 $2^{\lfloor \mathrm{lb}\, l_{\max,1} \rfloor}$b. 如图 2 所示,一个布隆过滤器通过 2 次自我复制将自身扩展了 4 倍.经过这样的扩展,一条原本曾在该布隆过滤器上置位的元素在扩展后的过滤器上经过 Hash 函数映射,其对应的 k 位依旧全部为 1,因此扩展操作不会影响布隆过滤器的正常使用.

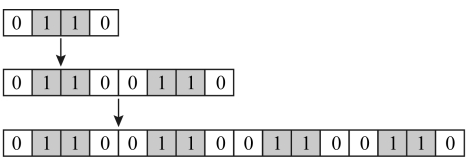


Fig. 2 Convert the bloom filter to the extended one by replicating itself twice

图 2 通过 2 次自我复制扩展布隆过滤器

第 1 轮筛选开始前,各监测器首先需要确定的是各自布隆过滤器的实际大小 $l_{i,1}^{*}$ 和 Hash 函数个数 k_1.因为第 1 轮筛选中各监测器并不需要使用同样大小的布隆过滤器,所以任一监测器不必知道其他监测器的记录情况,而只需要根据自身记录的元素数量确定对应的布隆过滤器参数即可.因此第 1 轮筛选的参数优化过程并不包含任何数据通信.

根据文献[15]提供的单个布隆过滤器最优参数的推导,监测器 M_i 可以计算得到对应布隆过滤器的最优参数,首先是位数组的理论大小 $l_{i,1}$:

$$l_{i,1}=-\frac{\ln \varepsilon^{*}}{(\ln 2)^{2}}\times N_{i,1}, \tag{1}$$

其中,ε^{*} 表示布隆过滤器的期望误报率,在前 2 轮筛选中它的值是预先给定的,所有监测器的 ε^{*} 都是统一的.$N_{i,1}$ 则代表第 1 轮筛选前监测器 M_i 记录的元素数量.为了后续布隆过滤器的扩展需要,监测器 M_i 对应布隆过滤器的实际大小 $l_{i,1}^{*}$ 应调整为

$$l_{i,1}^{*}=2^{\lfloor \mathrm{lb}\, l_{i,1} \rfloor}. \tag{2}$$

对于 Hash 函数个数 k_1,监测器同样依据文献[15]给出推导计算它的实际最优值:

$$k_1=\left\lfloor -\frac{\ln \varepsilon^{*}}{\ln 2} \right\rfloor, \tag{3}$$

由于 ε^{*} 采用统一的预设值,所有监测器求得的 k_1 也是一致的.

第 1 轮筛选会筛去大量的非 t-重复元素,使得后续轮次的筛选中各监测器元素数量不均衡的状况得到大幅度的改善.所以在此后的第 q 轮筛选中,机制会以满足剩余元素数量最多的监测器 $M_{\max,q}$ 的需求为标准,为各监测器设置统一的布隆过滤器大小 $l_{\max,q}$ 和 Hash 函数数量 k_q.因此 $l_{\max,q}$ 和 k_q 需由协调器在掌握了所有监测器的具体情况后计算得到并分配给各个监测器.

2.3 首轮筛选后的参数优化

第 2 轮筛选开始前,各监测器会先将各自列表中剩余元素的数量 $N_{i,2}$ 发送给协调器.协调器收到所有监测器信息后,会找出其中的最大值 $N_{\max,2}$,然后以 $N_{\max,2}$ 为基础为各监测器计算统一的 $l_{\max,2}$ 和 k_2.计算方式与第 1 轮筛选中类似:

$$l_{\max,2}=\left\lfloor -\frac{\ln \varepsilon^{*}}{(\ln 2)^{2}}\times N_{\max,2} \right\rfloor,$$

$$k_2=\left\lfloor -\frac{\ln \varepsilon^{*}}{\ln 2} \right\rfloor. \tag{4}$$

此后协调器会将 $l_{\max,2}$ 和 k_2 发送给各监测器,以供它们对布隆过滤器进行初始化.

一般情况下,前 2 轮筛选都能够大幅降低通信开销,所以不考虑其必要与否.而在后续的数轮筛选中,协调器就需要在筛选前判断该轮筛选的必要性以决定是否继续进行筛选了.

2.4 后续筛选的参数优化

相比于前 2 轮筛选,后续几轮的筛选从前次筛选中继承了各监测器对元素列表用布隆过滤器编码后产生的大小一致的位数组.借助这些数据,协调器可以在筛选开始前对元素的分布状况进行预估.记只出现在 m 个监测器列表中的元素数量为 f_m,在前次筛选整合出结果位数组前,协调器只要分别统计出逐位累加后各个位中数值等于 $0,1,\cdots,n$ 的位的数量,根据文献[17]中提出的持续流量估计算法,就能够以较高的准确率估计出 f_m 的值.由于原算

法是基于1次Hash置位的Bitmap进行的流量估计,而在本文中,输入是k_q次Hash置位的位数组,所以在计算出数值后还要将其除以k_q以得到对f_m的估计值.

对于只出现在m个监测器列表中的元素被记录下的次数,可以用$m\times f_m$进行计算.根据t-重复元素的定义,协调器能够估计出t-重复元素在所有监测器列表中数量的总和:

$$N^*=\sum_{m=t}^{n}m\times f_m. \tag{5}$$

根据布隆过滤器的相关理论,只要给定了待编码集合的大小以及期望的误报率阈值后,协调器就能依照公式计算出单个布隆过滤器的最优参数.但是单个布隆过滤器的最优并不等同于多个布隆过滤器整合后的全局最优,在本机制的筛选过程中需要考虑的是全局的筛选成本和效果.因此在具备估计不同出现次数的元素分布状况的条件后,本文给前2轮筛选外其他轮次的筛选设计了考虑全局最优的参数优化方案.

在第q轮筛选中,协调器在已知各监测器拥有的剩余元素数量的情况下,不仅要为所有布隆过滤器统一分配一个合理的$l_{\max,q}$和k_q,还要判断该轮筛选是否有必要进行.为此,本文为第q轮筛选构造了一个收益函数p_q:

$$p_q=\Big((1-\varepsilon_q)\times\sum_{i=1}^{n}r_{i,q}-N^*\Big)\times h-2\times n\times l_{\max,q}, \tag{6}$$

其中,h表示直接发送1条元素的平均通信开销,ε_q表示第q轮筛选的理论误报率,$r_{i,q}$表示第q轮筛选开始前监测器M_i剩余的元素总数,由监测器M_i发送给协调器.在第q轮筛选前,整个监测系统中所有待筛选的元素总数为$\sum_{i=1}^{n}r_{i,q}$.筛选后,系统中剩余元素可以分为误报元素和t-重复元素2类,其中误报元素的数量可以用$\varepsilon_q\times\sum_{i=1}^{n}r_{i,q}$估计,$t$-重复元素的数量则为用前轮筛选的结果估计出的$N^*$.综上所述,协调器可以用$(1-\varepsilon_q)\times\sum_{i=1}^{n}r_{i,q}-N^*$估计第$q$轮筛选筛去的元素总数.对于系统而言,若没有进行第$q$轮筛选,则这部分被筛去的元素会分别被各个监测器以每条h(单位为b)的通信消耗直接发送给协调器,因此,发送这些元素所传输的数据量即是该轮筛选在减少通信开销上的贡献.此外,筛选本身所产生的通信开销也必须纳入考虑.筛选过程中包含2次数据交互,无论是监测器发送的编码后的位数组,还是协调器发送的结果位数组,其数量都为n,且长度都为$l_{\max,q}$,所以本轮筛选的开销为$2\times n\times l_{\max,q}$(单位为b).用第$q$轮筛选所节省的数据传输量减去筛选消耗的通信资源,即为该轮筛选为整个探测过程减小的通信开销.这也是本文为第q轮筛选构造的收益函数p_q所包含的物理意义.

在定义了第q轮筛选的收益后,本轮的优化目标自然是将收益最大化.由于最大化p_q隐含了用尽可能少的通信开销筛去尽量多的非t-重复元素,因此在优化过程中不必要对筛选的效果做额外的约束,可以将第q轮筛选的优化方程规约为

$$\max\ p_q \quad \text{s.t. } 1\leqslant k_q\leqslant l_{\max,q}, \tag{7}$$

其中,ε_q表示第q轮筛选的理论误报率.文献[17]中有类似的推导过程,只需要将由置位1次的Bitmap扩展到置位k次的布隆过滤器即可,此处直接给出推导结果:

$$\varepsilon_q=\Big(\sum_{i=t}^{n}C_n^i\times P_i\Big)^{k_q}. \tag{8}$$

其中,P_i可令i从0取到n依次计算得到:

$$P_i=\left(1-\frac{1}{l_{\max,q}}\right)^{k_q\sum_{j=i+1}^{n}f_j}\times\prod_{j=1}^{i}\left(1-\frac{1}{l_{\max,q}}\frac{C_n^j-C_i^j}{C_n^j}\right)^{k_qf_j}-\sum_{j=0}^{i-1}C_i^jP_j. \tag{9}$$

将式(6)(8)(9)代入式(7)中,求解优化问题,协调器可以得到k_q和$l_{\max,q}$的值,同时也能够解出本轮收益p_q.在第q轮筛选的参数优化中,文献[17]提出的持续流量估计算法的时间复杂度与空间复杂度为$O(N)$,N为第q轮筛选前系统中的元素总数.求解优化问题则需要遍历每一对可能的$(l_{\max,q},k_q)$组合并计算对应的收益p_q以找出最优解,其时间复杂度为$O(L_{\max,q}\times K_{\max,q})$,空间复杂度为$O(1)$,其中$L_{\max,q}$和$K_{\max,q}$分别表示第$q$轮筛选中$l_{\max,q}$和$k_q$可能的最大值.综上,第$q$轮筛选的参数优化算法的时间复杂度为$O(N+L_{\max,q}\times K_{\max,q})$,空间复杂度为$O(N)$.

根据p_q的定义,当$p_q\leqslant 0$时,本轮筛选无法给整个探测过程带来任何收益,甚至会提高总体的通信开销.此时将参数发送给监测器,继续第q轮的筛选显然是不合适的,协调器将依据p_q的值决定筛选是否继续.筛选过程会耗费一定的时间成本,所以当收益不大的时候,筛选带来的通信开销的降低可能不足以弥补其所消耗的时间.因此,可以根据具体

应用场景的要求和系统的规模为收益设置一个阈值 λ,当收益$\leqslant\lambda$ 时,则认为没有继续下一轮筛选的必要.

为了防止对 f_m 和ε_q 的估计误差影响机制正常收敛,在判断第 q 轮筛选是否有必要进行时,除了对第 q 轮筛选的预期收益 p_q 进行估计判断外,协调器还会对第 $q-1$ 轮筛选产生的实际收益 p^*_{q-1}进行验证,检验其是否达到预期收益 λ.p^*_{q-1}的计算为

$$p^*_{q-1}=h\times\sum_{i=1}^{n}(r_{i,q-1}-r_{i,q})-2\times\sum_{i=1}^{n}l^*_{i,q-1},\quad(10)$$

其中,$\sum_{i=1}^{n}(r_{i,q-1}-r_{i,q})$ 表示第 $q-1$ 轮筛选实际筛去的元素总数,$l^*_{i,q-1}$表示第 i 个监测器在第 $q-1$ 轮筛选中所用布隆过滤器的大小,$2\times\sum_{i=1}^{n}l^*_{i,q-1}$ 则为第 $q-1$ 轮筛选所需的通信开销.通过计算第 $q-1$ 轮筛选筛去的那部分元素的直接通信代价和筛选过程中产生的通信开销的差值,即可求出其实际收益.

只有当同时满足 $p^*_{q-1}>\lambda$ 和 $p_q>\lambda$ 这 2 个条件时,第 q 轮筛选才会继续,否则协调器将发出终止筛选的信号.因为机制所能取得的总收益是有限的,因此在保证每轮筛选的上一轮筛选都能取得正收益的情况下,机制是可以在有限轮筛选后收敛的,且除最后一轮筛选外的其他轮次的收益都是绝对符合预期的.具体算法描述如算法 1 所示:

算法 1. 第 q 轮参数优化算法($q\geqslant3$).

输入:各监测器持有元素数目集合$\{N_{i,q}\}_{i\in[1,n]}$、估计的 t-重复元素数目 N^*、收益阈值 λ;

输出:统一的布隆过滤器的大小 $l_{\max,q}$、Hash 函数数量 k_q 或终止筛选的信号.

① 据文献[17]的算法估计$\{f_1,f_2,\cdots,f_n\}$;
② 据式(5)估计 N^* 并更新;
③ 据式(6)(8)确定 p_q,ε_q;
④ 据式(7)解出 $l_{\max,q}$,k_q 并计算 p_q;
⑤ 据式(10)计算 p^*_{q-1};
⑥ if $p_q>\lambda$ and $p^*_{q-1}>\lambda$
⑦ return $l_{\max,q}$,k_q;
⑧ else
⑨ return 终止信号;
⑩ end if

2.5 机制分析

本文提出的 t-重复元素检测机制最重要的目标是用尽可能少的通信开销完整、无误报地找出单个监测周期中系统内所有的 t-重复元素.基于这一目的,上文为 t-重复元素检测问题制定了 3 个设计需求,接下来将证明本机制是满足这 3 点需求的.

定理 1. 所设计的 t-重复元素检测机制可以保证探测结果的完整性.

监测器主要依靠布隆过滤器进行数据压缩.在任意的第 q 轮筛选中,根据布隆过滤器的运作原理,任意一条 t-重复元素在其被记录的至少 t 个监测器的布隆过滤器上都会被映射到相同的 k_q 位,且这些位置都将被置 1.因此数据整合后其对应在结果位数组上的 k_q 位也将全部被置为 1,当这些记录了它的监测器查验自身元素列表时,该条元素会被保留.所以筛选过程中可以确保不会剔除任何一条 t-重复元素.而最终协调器整合元素数据的过程中,对于这些被至少 t 个监测器传输来的元素,协调器不会有任何检测缺漏的情况,即机制可以完整地检测出所有 t-重复元素.

定理 2. 所设计的 t-重复元素检测机制可以保证探测结果的无误报.

对于监测器元素列表中存在的非 t-重复元素,虽然多轮次的筛选无法保证将其全部筛去,但可以证明它们并不会影响探测结果的准确性.在最后协调器的整合步骤中,各监测器将包含了少量非 t-重复元素和全部 t-重复元素的元素集合传输给协调器.通过比对所有接收到的元素集合,协调器可以找到所有的 t-重复元素.而对于非 t-重复元素,以监测器 M_i 中的某条非 t-重复元素为例,当且仅当其本就在监测器 M_i 的列表中且通过了多次筛选,它才会最终被发送给协调器.假设它被协调器误报为 t-重复元素,则它必须出现在至少 t 个监测器的传输列表中,即它在本周期内确实地被 t 个或更多的监测器记录了下来,那么按照定义,它应当是一条 t-重复元素,因此假设并不成立,本文提出的机制可以满足探测结果无误报的要求.

定理 3. 所设计的 t-重复元素检测机制可以保证每一轮筛选的必要性,从而最大程度降低通信成本.

在满足了结果准确性需求的前提下,机制的性能也需要得到保证.本文给出的机制综合了 t 的取值以及所有监测器拥有元素的数目等因素从全局最优的角度对各监测器对应的布隆过滤器进行了参数优化,因此无论元素在各监测器列表中的分布状况如何,机制都能保持良好的性能.而筛选终止条件的判断则确保了每一轮筛选都是能够取得收益的必要筛选,使得通信开销永远随着筛选轮次数的递增而递减,因此可以最大程度地降低通信成本.

3　实验分析

本节给出仿真实验的具体细节,结合对比实验分析了机制在降低通信开销上的效果,并验证了机制在不同元素分布状况下的性能.

3.1　实验设置

本文实验采用了公开数据集 CAIDA 中的真实网络流量数据作为数据集,取其中 12 min 内服务器记录下的 IP 访问信息,将其划分为 12 个时间片.实验中,每个时间片(1 min)的数据都会被模拟为分布式监测系统中一个监测器节点所记录的元素列表.在模拟的系统中一共有 4 100 541 条元素.其中每条元素包含 1 个源地址和 1 个目的地址,大部分 IP 地址都是 IPv4 地址,只有极少量的 IPv6 地址,所以直接发送 1 条元素的通信开销 h 在本次实验中被设置为 64 b.在系统包含的 12 个监测器节点中,元素的分布大致均匀,平均每个监测器约有 34 万条元素记录.为了适应各种应用场景,本文将 t-重复元素的阈值 t 的数值设为 4～12 之间的任意数值,设收益阈值 $\lambda=0$ 并分别进行实验,实验结果均为多次实验后得到的平均结果.

本次实验的实验环境为 Intel® Core™ i7-7700 CPU 3.60 GHz,16 GB RAM.

3.2　性能分析

为了用作对比实验,本文将文献[10]中提出的 CFSP 算法加以修改,在整合压缩数据时用逐位累加的方法取代逐位与,并在筛选结束后将剩余元素全部发送给协调器,以适应所要解决的 t-重复元素检测问题.此外,虽然 DISPERSE 算法存在误报与漏报情况,本文同样将其与直观算法一起纳入对比.实验将从通信开销、时间开销、误报漏报、算法的鲁棒性等方面对机制进行评价.

4 种算法的通信开销如图 3 所示,随着阈值 t 的增大,改进后的 CFSP 算法和本文提出的机制所耗费的通信开销不断降低,DISPERSE 算法的通信开销基本趋于稳定,直观算法的通信开销则始终保持在一个定值.这是因为,t 值的增大意味着 t-重复元素的减少,所以在筛选过程中可以筛去更多的无关元素,进一步降低了整体的通信开销.而无论 t 取何值,DISPERSE 算法都需对所有元素进行编码并且只依赖 1 次通信获取结果,因此开销趋于稳定.直观算法中监测器总是将所有记录下的元素直接发送给协调器,其通信开销也是固定的,与 t 的取值无关.可以看到,无论 t 取何值,所提出的机制总能在通信开销的降低上取得较高的收益.当 $t>8$ 时,机制的收益优于其他 3 种算法.根据 t 的不同取值,本机制相比于直观算法可以降低 14%～79%的通信开销,相比于改进后的 CFSP 算法可以降低 8%～26%的通信开销,$t>8$ 时与 DISPERSE 算法比较可以节省 11%～42%的通信开销.机制会持续 3～4 轮的筛选,相较于改进后的 CFSP 算法节省了 3～4 个轮次.

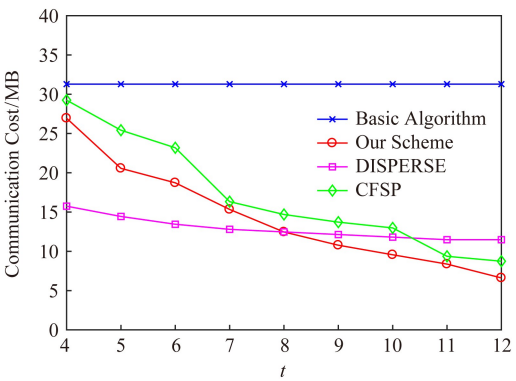


Fig. 3　Communication cost of our scheme and other three algorithms using the original data

图 3　原始数据下机制与 3 种算法的通信开销对比

因为 DISPERSE 算法的特殊性,表 1 单独列出了其与本机制的性能数据以作对比,包含了全局的通信开销、误报率和漏报率.表 1 中的 FPR(false positive rate)代表检测结果中被误判为 t-重复元素的非 t-重复元素在所有非 t-重复元素中所占的比例,即误报率.而 FNR(false negative rate)代表未被检测出的 t-重复元素在所有 t-重复元素中所占的比例,即漏报率.从表 1 可以看出,随着 t 值的不断增大,2 种算法的通信开销都呈下降趋势.但相比之下,本机制的通信开销下降速度更快,在 $t=8$ 时基本与 DISPERSE 算法持平,此后的通信开销更是优于 DISPERSE 算法.此外,本机制在实验中始终没有出现误报漏报现象,而 DISPERSE 算法在保持着极低的误报率的情况下,其漏报率却不容忽视,且漏报率会随着 t 值的增大而增大,从 14%～38%不等.因此,相比于 DISPERSE 算法,本机制具有更为优越的性能.

由于实验使用了连续时间片的网络流量数据对监测器的元素记录进行模拟,元素在系统中分布得较为均匀.为了检测机制的鲁棒性,我们进一步地对数据进行一定的处理,为每一个时间片随机初始化一个概率并依据该概率在该时间片数据中随机选取

一些数据丢弃.经过处理后,每个时间片包含的数据量从2～27万条不等,由此模拟出的分布式监测系统中,元素在各监测器中的分布状况极不均衡.在此基础上,实验结果如图4所示,可以看到,即使在元素分布不均匀的状况下,本机制依旧可以保持良好的性能,甚至比在元素分布均匀的系统中表现得更为优越.而当 t 的取值较小时,CFSP算法则表现得有些不稳定,$t=4$ 时其消耗的通信成本甚至超过了直观算法.机制相较直观算法可以节省28%～96%的通信开销,相较改进后的CFSP算法可以节省21%～47%的通信开销,$t>4$ 时与DISPERSE算法比较可以节省9%～91%的通信开销.

Table 1 The Performance Comparison of Our Scheme and DISPERSE Using the Original Data

表1 原始数据下所提出的机制与DISPERSE算法的性能比较

t	Our Scheme			DISPERSE		
	Communication Cost/MB	*FPR*	*FNR*	Communication Cost/MB	*FPR*	*FNR*
4	26.95	0	0	15.75	7E−6	0.14
5	20.57	0	0	14.44	4E−6	0.15
6	18.72	0	0	13.45	3E−6	0.17
7	15.34	0	0	12.80	2E−6	0.20
8	12.46	0	0	12.47	2E−6	0.23
9	10.78	0	0	12.14	2E−6	0.26
10	9.56	0	0	11.81	1E−6	0.30
11	8.38	0	0	11.48	1E−6	0.33
12	6.61	0	0	11.48	1E−6	0.38

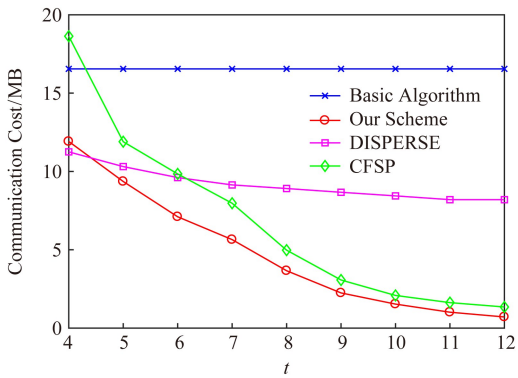


Fig. 4 Communication cost of our scheme and other three algorithms using the processed data

图4 处理后数据下机制与3种算法的通信开销对比

在实验中,根据 t 的不同取值,筛选会持续3～4个轮次.其中当 $t=8$ 时各轮筛选的表现情况如表2所示,系统总共进行了3轮筛选即达到终止条件,在此基础上又额外进行了第4轮筛选以作比较.

从表2可以看出,随着筛选轮次的增加,筛选的收益在不断减小.这是因为筛选降低的通信开销和该轮筛选滤去的元素数量呈正相关,和每条元素的平均过滤代价呈负相关.伴随着筛选的进行,非 t-重复元素的数量大幅度减少,筛选中可以滤去的元素数也随之减小.而压缩过程中用于给 t-重复元素编码的那部分数据大小基本稳定,且在筛选耗费的总成本中的占比不断增加,这使得筛选成本的降低幅度远远比不上筛去元素的减少幅度.因此随着筛选次数的增加,筛选的收益,即其降低的整体通信开销不断减小.到了第4轮筛选,一共只筛去了48 635条元素,平均筛去1条元素需要108.5 b,而直接传输1条元素仅需64 b,所以该轮筛选反而让整体收益有所降低.

Table 2 Performance of Each Round Using the Processed Data when $t=8$

表2 当 $t=8$ 时处理后数据下各轮筛选的表现情况

Round	Rest Elements	Filtered Elements	Total Communication Cost/MB	Communication Cost Reduction/MB	Time Cost in Monitor/s	Time Cost in Coordinator/s
1	2 168 914	698 560	11.87	4.68	0.64	0.86
2	1 470 354	971 837	5.23	6.64	0.43	0.70
3	498 517	346 812	3.56	1.68	0.17	1.61
Extra 4	151 705	48 635	3.81	−0.26		

此外,实验还从时间开销的角度对机制进行了性能分析.本机制的时间开销主要可分为通信和运算2部分.其中通信时间与通信开销成正比,比如在100 Mbps的网络环境下,无论t取何值,机制都能在1～2 s内完成所有的数据传输.即使在50 Mbps的网络带宽下,机制也能在3 s内完成通信.接下来主要对运算时间进行分析,其结果如表2所示.依照机制的检测流程,本文将一轮筛选的运算时间划分为监测器端和协调器端2部分,协调器端的运算时间主要用于参数优化和整合位数组,监测器端的运算时间则包含了元素置位和筛去非t-重复元素所耗费的时间.其中,前2轮筛选的参数优化只进行了简单的计算,所以几乎不占用时间资源.而监测器进行置位和筛选是所有监测器同步进行的,所以只取这些监测器中运算时间的最大值.可以看到,随着筛选轮次的增加,元素列表的规模不断减小,监测器端所消耗的运算时间也呈下降趋势.在第3轮及后续轮次的筛选中,因为需要综合全局状况求解最优参数,所以相比前2轮筛选依据公式计算参数,协调器端需要更多的运算时间.

上述实验过程中,协调器总是依据对下一轮收益p_q的预测未达到预期而发出终止信号,每一轮筛选都能有效降低整体的通信开销.而协调器给出的检测结果包含了所有t-重复元素,且没有任何一条非t-重复元素被误报,这也验证了此前的理论分析.

4　总　　结

针对分布式系统中t-重复元素检测问题,以最大化降低通信开销为优化目标,提出了一种适用于分布式监测系统的t-重复元素检测机制.通过筛选过程中对监测器信息的整合和动态预测,确保了机制的强健性,保证了每一轮筛选通信代价的最小化.使用真实网络流量数据进行的模拟实验结果表明:无论系统中元素分布均匀与否,机制都能取得良好的效果.在相似的性能及实验环境下,机制的稳定性和在降低通信开销上的表现优于直观算法,改进后的CFSP算法和DISPERSE算法.

参　考　文　献

[1] Spitzner L. The honeynet project: Trapping the hackers [J]. IEEE Security & Privacy, 2003, 1(2): 15-23

[2] Guo Junquan, Zhuge Jianwei, Sun Donghong, et al. Spampot: A spam capture system based on distributed honeypot [J]. Journal of Computer Research and Development, 2014, 51(5): 1071-1080 (in Chinese)
(郭军权, 诸葛建伟, 孙东红, 等. Spampot: 基于分布式蜜罐的垃圾邮件捕获系统[J]. 计算机研究与发展, 2014, 51(5): 1071-1080)

[3] Ginsberg J, Mohebbi M H, Patel R S, et al. Detecting influenza epidemics using search engine query data [J]. Nature, 2009, 457(7232): 1012-1014

[4] Manjhi A, Shkapenyuk V, Dhamdhere K, et al. Finding (recently) frequent items in distributed data streams [C] // Proc of the 21st Int Conf on Data Engineering. Piscataway, NJ: IEEE, 2005: 767-778

[5] Cao Pei, Wang Zhe. Efficient top-k query calculation in distributed networks [C] //Proc of the 23rd Annual ACM Symp on Principles of Distributed Computing. New York: ACM, 2004: 206-215

[6] Babcock B, Olston C. Distributed top-k monitoring [C] // Proc of the 2003 ACM SIGMOD Int Conf on Management of Data. New York: ACM, 2003: 28-39

[7] Li Mei, Lee W C. Identifying frequent items in P2P systems [C] //Proc of the 28th Int Conf on Distributed Computing Systems. Piscataway, NJ: IEEE, 2008: 36-44

[8] Lahiri B, Tirthapura S. Identifying frequent items in a network using gossip [J]. Journal of Parallel and Distributed Computing, 2010, 70(12): 1241-1253

[9] Eppstein D, Goodrich M T, Uyeda F, et al. What's the difference?: Efficient set reconciliation without prior context [J]. ACM SIGCOMM Computer Communication Review, 2011, 41(4): 218-229

[10] Cai Zhiping, Chen Min, Chen Shigang, et al. Searching for widespread events in large networked systems by cooperative monitoring [C] //Proc of the 23rd Int Conf on Network Protocols. Piscataway, NJ: IEEE, 2015: 123-133

[11] Michael L, Nejdl W, Papapetrou O, et al. Improving distributed join efficiency with extended Bloom filter operations [C] //Proc of the 21st Int Conf on Advanced Information Networking and Applications. Piscataway, NJ: IEEE, 2007: 187-194

[12] Ramesh S, Papapetrou O, Siberski W. Optimizing distributed joins with Bloom filters [C] //Proc of the Int Conf on Distributed Computing and Internet Technology. Berlin: Springer, 2008: 145-156

[13] Chen Jiyu, Cai Zhiping, Chen Shiping. Efficient distributed joint detection of widespread events in large networked systems [C] //Proc of the 2016 IEEE Global Communications Conf. Piscataway, NJ: IEEE, 2016: 1-6

[14] Jeong J, Naqvi S M A, Yoon M. Accurate and communication-efficient detection of widespread events [J]. IEEE Access, 2018, 6: 61728-61734

[15] Bloom B H. Space/time trade-offs in Hash coding with allowable errors [J]. Communications of the ACM, 1970, 13(7): 422-426

[16] Dai Haipeng, Li Meng, Liu A. Finding persistent items in distributed datasets [C] //Proc of the 2018 IEEE Conf on Computer Communications (INFOCOM). Piscataway, NJ: IEEE, 2018: 1403-1411

[17] Huang He, Sun Yu'e, Chen Shigang, et al. You can drop but you can't hide: k-persistent spread estimation in high-speed networks [C] //Proc of the 2018 IEEE Conf on Computer Communications (INFOCOM). Piscataway, NJ: IEEE, 2018: 1889-1897

Lu Le, born in 1995. Master candidate at the School of Computer Science and Technology, Soochow University, China. His main research interest is data integration in distributed system.

Sun Yu'e, born in 1983. Associate professor at the School of Rail Transportation, Soochow University, China. Member of the IEEE, the ACM and the CCF. Her main research interests include network traffic measurement, spectrum auction, privacy preserving, and wireless networks.

Huang He, born in 1983. Professor at the School of Computer Science and Technology, Soochow University, China. Member of the IEEE, the ACM and the CCF. His main research interests include network traffic measurement, spectrum auction, privacy preserving, crowdsourcing, software-defined networks, and satellite networks. (huangh@suda.edu.cn)

Wang Runzhi, born in 1995. Master candidate at the School of Computer Science and Technology, Soochow University, China. Her main research interest is truth discovery. (20174227037@stu.suda.edu.cn)

Cao Zhen, born in 1996. Master candidate at the School of Computer Science and Technology, Soochow University, China. His main research interest is network traffic measurement. (zcao96@stu.suda.edu.cn)