ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.21, No.5, May 2010, pp.1115−1126
doi: 10.3724/SP.J.1001.2010.03568

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

一种基于双重 Counter Bloom Filter 的长流识别算法*

吴 桦[1,2+], 龚 俭[1,2], 杨 望[1,2]

[1](东南大学 计算机科学与工程学院,江苏 南京 210096)

[2](江苏省计算机网络技术重点实验室,江苏 南京 210096)

Algorithm Based on Double Counter Bloom Filter for Large Flows Identification

WU Hua[1,2+], GONG Jian[1,2], YANG Wang[1,2]

[1](School of Computer Science and Engineering, Southeast University, Nanjing 210096, China)

[2](Key Laboratory of Computer Network Technology of Jiangsu Province, Nanjing 210096, China)

+ Corresponding author: E-mail: hwu@njnet.edu.cn, http://www.seu.edu.cn

Wu H, Gong J, Yang W. Algorithm based on double counter bloom filter for large flows identification. *Journal of Software*, 2010,21(5):1115−1126. http://www.jos.org.cn/1000-9825/3568.htm

Abstract: An algorithm based on double counter bloom filter for long flows identification (CCBF) is proposed in this paper. Double counter bloom filter structure is used to distinguish the process of the long flow filtration from the long flow existence. The false positive rate of the algorithm is analyzed. The relationship of the memory requirement and the error rate is analyzed through simulation. It is shown that with the same restriction of the memory resource, the average error of this algorithm is less than the existing similar algorithms. The analysis of the time performance shows this algorithm is capable of dealing with traffic up to 1 500kpps.The results reflect that this algorithm can be used to monitor the long flows on backbone network.

Key words: large flow identification; double; counter bloom filter; network measurement; backbone network

摘 要: 提出了一种基于双层 Counter Bloom Filter 的长流识别算法(algorithm based on double counter bloom filter for long flows identification,简称 CCBF).该算法使用两层 Counter Bloom Filter 结构,将长流过滤和长流存在分开处理.分析了该算法的误判率,通过模拟数据分析了算法错误率和内存资源限制的关系,并在相同内存资源限制的条件下,将该算法与类似算法的准确性进行了比较.结果表明,在数据量较大的情况下,该算法具有比现有算法更小的平均错误率;对算法的时间效率分析表明,该算法可以达到 1 500kpps 的处理速度.各项指标反映出,该算法可以应用于大规模主干网的长流监测.

关键词: 长流识别;双重;计数 Bloom Filter;网络测量;主干网

中图法分类号: TP393 **文献标识码**: A

网络测量是网络管理的基础.随着网络规模的日益扩大和网络流量的急速增加,对网络行为的精确测量越来越困难.流测量作为满足特定属性网络报文的一种聚类方法,既可以满足细粒度网络管理的需求,在数据存储

* Supported by the National Basic Research Program of China under Grant Nos.2003CB304804, 2009CB320505 (国家重点基础研究发展计划(973)); the National Key Technology R&D Program of China under Grant No.2008BAH37B04 (国家科技支撑计划)

Received 2008-04-29; Accepted 2008-12-29

和处理开销上也具有较好的优势,因此在网络管理领域具有很好的应用前景.

由于受网络设备的资源限制,在主干网中将流信息完整地采集下来供分析用,在存储和分析的开销上是不可行的.网络流测量系统必须按照一定的原则有选择地收集全部流量信息的子集.目前,对网络流测量的研究根据测量对象的不同,主要分为以下 3 个方面:

(1) 全状态流测量:测量系统按照定义格式维护所有活动 IP 流的状态信息,包括 IP 流规范信息(如五元组表示,包括源宿 IP 地址、源宿端口和协议类型)、IP 流的起始时间、IP 流流长度信息(双向报文数、字节数)等等.如 Cisco 公司基于其硬件路由器产品的 NetFlow[1],由于其标准的开放性而被业界厂商广泛支持,其兼容协议形式在各家硬件路由器产品多有实现.由于硬件资源的限制,这类产品为控制进入测量系统的报文数,都引入抽样作为解决方案.但是,抽样带来的问题是对原始流信息估计的准确性.相关的研究有:文献[2]研究了基于时间和基于报文到达次序为抽样的激发机制,分析系统抽样、随机抽样分层的测量技术;IETF(Internet engineering task force)的报文抽样测量工作组(PSAMP)[3]专门研究报文抽样测量的相关技术.抽样率、抽样方法和估计准确性是这类流测量系统的研究热点.

(2) 无状态维护流测量:这类测量系统不维护任何 IP 流的具体信息,只从宏观的角度对 IP 流流长分布、到达率等相关情况作简单的统计.这些统计信息可以提供网络的使用形式特征,帮助检测各种类型的网络安全事件,如 DDoS 攻击和网络蠕虫都会导致流长分布的变化.相关的研究有:Estan 等人[4]使用哈希技术给出了一族 Bitmap 算法来统计活动 IP 流的数量;Kumar 等人[5]使用贝叶斯统计估计流的大小分布.这一类研究使用资源很小,所获得的信息量也较少,不能满足网络管理和计费的要求.

(3) 部分状态维护流测量:这类测量系统有选择地挑选感兴趣的 IP 流维护其相关信息,但对其他 IP 流只做简单计数或者直接丢弃.已有多个文献(如文献[6,7])得到了 IP 流长分布符合重尾分布的结论.鉴于长流对网络管理和应用的重要作用,很多算法的目的是尽量精确地记录长流记录而忽略短流,这类算法称为长流识别算法.本文第 1 节着重介绍了这类算法的研究现状.

本文提出了一种基于双重 Counter Bloom Filter 的长流识别算法(algorithm based on double counter bloom filter for long flows identification,简称 CCBF).第 1 节介绍长流识别算法的研究现状.第 2 节首先介绍 CCBF 算法的基础 Counter Bloom Filter,然后介绍 CCBF 算法设计.第 3 节对 CCBF 算法的误判率进行理论分析,用模拟方法验证内存大小对准确性的影响;将 CCBF 的错误率与类似算法进行比较并分析其时间效率.第 4 节是本文的总结.

1 长流识别算法研究现状

在不同的网络管理和监测应用中,流的长度是应用类型识别的一个主要依据,长流和短流的定义成为这些识别算法的基础.事实上,长流和短流并没有明确的界限,针对不同的应用需求,分隔点是不同的.一般认为,长流是在单位时间内发送的数据量在该网络总数据流量中占据一个较大的比例,或者该流的带宽资源占用率较大(如超过总带宽的 1%[8]).就一个测量系统来说,对长流的定义也需要是可调的,以适应不同的应用需求.在本文中,我们将长流定义为在某个时间段内到达报文数目超过某个特定值的流.因此,本文后面提到的长流和短流都是相对的.

由于 IP 流长分布具有显著的重尾分布特点,一般的随机抽样是偏向于长流的.在此基础上,改进的抽样算法增加了长流被抽中的可能性.Kodialam 等人[9]提出了一种简单、有效,偏向长流的抽样流采集方法.当一个流有两个报文一前一后被抽中之后才建立它的记录,因此,缓存中有一个寄存器保存上一个报文的索引,并与当前被抽中的报文比较,匹配之后才建立流记录(第 1 次出现)或更新流记录.相对于一般的随机抽样,长流被抽中的可能性更大.Duffield 等人[10]提出根据流长进行不等概率抽样.设流长为 x,相对应的抽样率为 $p(x)$,抽样率应该随着 x 的增加而增长,设 z 为以报文或者字节计算的流长度,他们选择 $p_z(x)=\min\{1,x/z\}$,这样,流长大于 z 的流全部被抽中,流长小于 z 的流按照和流长成正比的比率被抽中.在实际使用中,可以设定 L 为流量级别,只有当流量超过 L 时才计费,使得测量对小流量的误差不敏感.但是,由于流长使用的是估计值,流长估计的误差会导致测量结果

不正确.因此,如果将该方法应用于计费系统,则需要使用一定的方法进行调整.这些方法是从抽样的角度选择偏向长流的报文,建立起长流记录,必然存在统计学上的误差.

Estan 在文献[8]中给出检测大流的两种经典算法:Sample and Hold 以及 Multistage Filters(以下称 MF)算法,目的都是在存储资源有限的情况下识别出大流,只保存大流的流记录信息.

Sample and Hold 的基本原理是:由于网络流的重尾特性,大流包含的报文数目多,按一定的概率进行采样得到的结果必定是偏向大流的.当某个报文到达后,先看该报文标识的流记录是否存在:如果存在,则对其进行更新;如果不存在,则按一定的概率决定是否为该报文建立新的流记录.该方法同样由于采用了抽样方法会有一定的误差.

MF 采用的是“过滤器(filter)”方法,保证过滤器能够把所有的大流过滤出来.为减少 Hash 冲突造成的误判,Estan 设计了采用多个过滤器的方法,使用多个不同的 Hash 空间和 Hash 函数,在多处进行累加.当一个报文到达时,只有当它在多个 Hash 表对应的多个地址中的累积值都超过阈值时,才会被认为是一个大流被记录下来.通过这种方法可以极大地降低该方法的误判率.

Estan 对 Sample and Hold,MF 和 Sampled Netflow 这 3 种流采集算法进行了正确性比较,得出的结论是,对于极大流(在一个测量间隔内单个流传输量超过总体传输量 0.1%上的流),这 3 个算法都可以较好地识别出;但是对于相对来说较小的“大流”(在一个测量间隔内传输量小于总传输量的 0.01%的流),MF 算法精度更高.并且随着这个流定义的逐渐变小的趋势,精度上的差别越来越大.

王洪波等人在文献[11]中提出将最近最久未使用(least recently used,简称 LRU)算法应用到大流检测中.该算法本来是用于虚拟内存管理等应用中,若应用到长流识别则存在明显的问题.内存管理中并不需要保证所有的访问记录被保存,只需对最近使用的内存进行标记排序,需要处理的数据量较小;而长流识别中所有历史记录都需要参与排序,否则会导致计数发生错误.由于网络流量数目巨大,在排序和查找过程中,相应的时间和空间复杂性相当大,导致该算法无法适用于大规模主干网的流量检测.

目前为止,由于受设备资源等因素的限制,主干网中长流测量方法仍然是一个尚未得到理想解决的问题.本文提出了一种基于双重 Counter Bloom Filter 的长流识别算法(CCBF).该算法利用了网络流长分布的重尾分布特点,结构合理,实现简单.由于 Estan 在文献[8]中验证了 MF 算法相对于其他类似算法具有较高的精度,本文将 CCBF 算法的准确性与 MF 进行比较.得到的结论是,由于将长流过滤和长流存在分开处理,在相同的内存资源限制下,CCBF 算法的精确性优于 MF.各项指标表明,CCBF 算法可以应用于大规模主干网上的网络管理和监测.

2 CCBF 算法

2.1 Bloom filter

Bloom Filter[12]是一种空间效率很高的随机数据结构,它利用向量 V 很简洁地表示一个集合,并能判断一个元素是否属于这个集合.

在 Bloom Filter 结构中,需要使用 k 个哈希函数 $h_1,h_2,\ldots,h_k$,其具体实施方法如图 1 所示,主要包括初始化(init)、元素插入(insert)和元素查询(search)这 3 个过程.

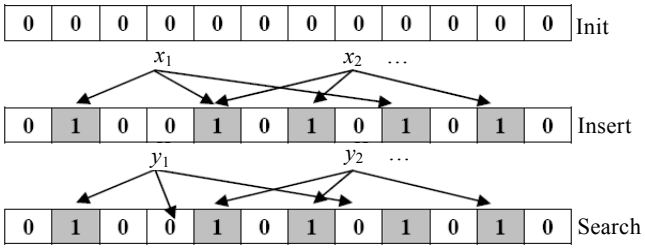


Fig.1 Demonstrations of bloom filter

图 1 Bloom Filter 示例

初始状态时,Bloom Filter 是一个包含 m 位的位数组,每一位都置为 0,为了表达 $S=\{x_1,x_2,\ldots,x_n\}$这样一个 n 个元素的集合,Bloom Filter 使用 k 个相互独立的哈希函数,它们分别将集合中的每个元素映射到{1,…,m}的范围中.对任意一个元素 x,第 i 个哈希函数映射到向量 V 的位置 $h_i(x)$就会被置为 1($1\le i\le k$).如果一个位置多次被置为 1,那么只有第 1 次会起作用,后面几次将没有任何效果.在图 1 的 Insert 过程中,k=3,且有两个哈希函数选中同一个位置.在判断 y 是否属于这个集合时,我们对 y 应用 k 次哈希函数,如果所有 $h_i(y)$的位置都是 1($1\le i\le k$),我们就认为 y 是集合中的元素,否则就认为 y 不是集合中的元素.图 1 的 Search 过程中,y_1 就不是集合中的元素.y_2 或者属于这个集合,或者刚好是一个误判(false positive).

标准的 Bloom Filter 是一种很简单的数据结构,它只支持插入和查找两种操作.因为它不支持删除操作,如果要表达的集合经常变动,标准 Bloom Filter 就无法完成了.

计数型 Bloom Filter 称为 Counter Bloom Filter,最早是由 Fan 等人在文献[13]中提出的,其所需初始条件与标准 Bloom Filter 的区别在于将向量 V 的每个单元由 1 比特扩展为一个计数单元.当集合中的一个元素 s_i 使用 k 个哈希函数映射到向量 V 中时,将映射所得的 k 个单元 $h_1(s_i),h_2(s_i),\ldots,h_k(s_i)$的计数值都增加 1.这样,CBF 不仅像标准 Bloom Filter 一样支持元素的插入和查询操作,而且支持元素的删除操作.当需要将一个元素 s_j 从 CBF 中删除时,如果其映射所得的 k 个单元 $h_1(s_j),h_2(s_j),\ldots,h_k(s_j)$的计数值都大于 0 则均减小 1,否则认为 $s_j\notin S$.

本文通过使用两个 Counter Bloom Filter 的组合实现了对长流的识别,其中一个 CBF 为长流记录识别空间,另一个 CBF 为过滤空间,通过两个 CBF 的有机结合完成长流识别.

2.2 Double Counter Bloom Filter

周明中等人在文献[14]中使用 CBF 结构来标识流存在.与其不同的是,CCBF 算法使用两个 CBF 结构担任不同的角色:一个用来记录长流的存在信息,称为 Longflow_CBF;另一个用来对长流进行过滤判断,称为 Filter_CBF.CCBF 算法将长流存在和长流的过滤分开标识,这样就避免了已经被判别出的长流对后继判断的影响,极大地提高了效率.

CCBF 算法的基本思路如图 2 所示,当一个报文到达时,将报文头部的流标识(协议类型、源地址、宿地址、源和宿的端口号组成的五元组)作为 Hash 函数的输入,得到的 k 个输出映射到 Longflow_CBF 中.如果判断这个报文属于已确定的长流,就将该长流记录更新.如果不属于现有的长流记录,则将刚才 Hash 的结果再映射到 Filter_CBF 中.在这个向量空间的 k 个计数器中,首先将各个计数器递增,然后判断是否为一个长流,如果各个计数器中的最小值达到某个阈值(依据对长流的定义),就建立一个新的流记录,并相应地将 filter_CBF 中对应的各个计数器减去阈值,在 long_CBF 中对应的 k 个计数器中都增加 1.

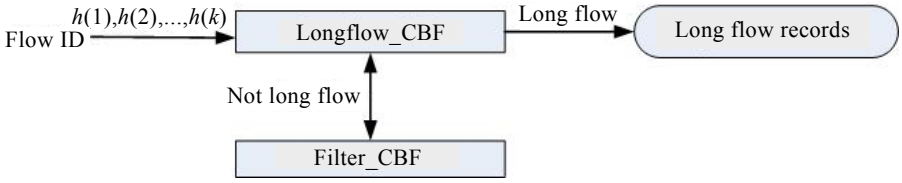


Fig.2 Structure of CCBF algorithm

图 2 CCBF 算法结构

算法的具体描述如下:

```
While a packet arrives
    calculate H=h(1),h(2),…,h(k)      //计算出 k 个地址
if H exist in Longflow_CBF            //属于已知长流
{       update Longflow_info          //修改这个流的记录
        If flow end                   //发现流结束标识
        {export flow record;
```

```
            decrease corresponding counters in Longflow_CBF;
            }
}
else                                    //不属于已知长流
{
increase corresponding counters in filter_CBF;
if min_counter>threshold{               //这个流的所有计数器都超过了阈值,将其从 filter_CBF 中的标识减
                                          去;并增加在 long_CBF 中的标识计数,增加新的长流记录
            decrease corresponding counters in filter_CBF;
            increase corresponding counters in Longflow_CBF;
            add new long flow info;
            }
}
end while
```

3 CCBF 算法分析

对长流识别算法的评价需要从算法的正确性、算法消耗的资源、算法执行的时间效率这 3 个方面进行.这几个方面是互相牵制并有所对立的.由于采集数据的最终目的是维护网络服务的正常运行,网络的运行成本是一个重要的因素,并非网络管理得越严、越细越好.在降低成本方面,不确定性和概率方法可起重要作用.对长流识别算法正确性的评价并不是所有符合要求的长流都被识别出,而是误判和漏判的概率是不是在可容忍范围内.相应地,资源消耗也是要求在一定的误差许可范围内,尽可能少地消耗系统资源.时间效率应该看每报文处理时间,对大规模主干网上的报文来说,时间效率是报文处理程序的关键.下面分别对这几个要素进行理论分析和实验比较.

3.1 误判率分析

正如所有使用 Bloom Filter 的应用中提到的,在能够容忍低错误率的应用场合下,Bloom Filter 通过极少的错误换取了存储空间的极大节省.由于 Bloom Filter 算法本身的特点,这种方法不会引起漏判,但是一个位置可以被置位多次.因为对一次查询来说,这些比特位的值可能是被其他元素置 1 的,这样会引起误判.在 Bloom Filter 中,如果 hash 函数产生的结果都是随机的,设 k 是 hash 函数的个数,m 是 bit 空间的大小,n 是需要判断的元素数目,则一个新的元素(例如 y)被识别为集合中元素的可能性(概率)为[15]

$$P_{bf}=\left(1-\left(1-\frac{1}{m}\right)^{kn}\right)^{k}\approx(1-\mathrm{e}^{-kn/m})^{k} \tag{1}$$

y 可能确实是属于该集合,也可能是一个误判(false positive).从原理上看,CBF 对一个新元素判断为已经存在的概率与 BF 是一样的,唯一的区别在于 CBF 可以执行减法操作,被减去的元素对误判的影响就被消除.对每一次判断来说,设从开始到这次判断有 i 个元素被加入又被删除,上面公式中的 n 为原来的 n 减去 i,即

$$P_{cbf}=\left(1-\left(1-\frac{1}{m}\right)^{kn}\right)^{k}\approx(1-\mathrm{e}^{-k(n-i)/m})^{k} \tag{2}$$

由此可以看出,在其他参数不变的条件下,CBF 误判率变小的原因相当于样本的总数变小了.上式中的 $n-i$ 在实现中实际上是 CBF 的一个计数器的瞬间值,这个值在有长流加入时加 1,在有长流结束时减 1.如果我们把公式(1)中的 n 看成是被标记元素的总数,那么 CBF 的误判率同样可以用公式(1)来标识,只是相对来说,在 BF 中,这个值是递增的,而在 CBF 中,这个值可增、可减.

在 CCBF 算法中,使用了两个 CBF,下面分别对 Longflow_CBF 和 Filter_CBF 进行分析.

Longflow_CBF 实际上是一个普通的 CBF,因此其误判率为

$$P_{longflow\_cbf}=\left(1-\left(1-\frac{1}{m}\right)^{kn}\right)^{k}\approx(1-\mathrm{e}^{-kn/m})^{k} \tag{3}$$

从公式(3)可以看出,在标识长流记录的 *Longflow_cbf* 中,影响误判率的因素有 3 个:已经被标识的长流个数 n、Hash 函数的个数 k、Hash 空间中计数器的大小 m.

Filter_CBF 的误判率就比较复杂了.在这个数据结构中,加入的元素是每个报文的流标识,需要判断的却是是否有长度超过阈值的长流存在.加入的元素和判断元素不是同一种类型,因此在数据的表示上虽然与 Counter Bloom Filter 相同,但是判别的类型却是不同的,误判率自然也不同.

假设需要识别的长流是报文数超过 j 的流,在 Filter_CBF 中,判断一个长流是否存在的依据就是在这个流标识经过 k 个 HASH 计算得到的计数器上,是不是所有的计数器的值都超过了阈值 j.类似于 CBF 的误判率,需要知道一个报文所代表的流标识属于一个长流的概率.

首先计算第 i 个 Counter 被增加 j 次的概率,其中,n 为集合元素个数,k 为哈希函数个数,m 为 Counter 个数(对应着原来位数组的大小):

$$P\{c(i)=j\}=\binom{nk}{j}\left(\frac{1}{m}\right)^{j}\left(1-\frac{1}{m}\right)^{nk-j} \tag{4}$$

那么第 i 个计数器大于等于 j 的概率为

$$P\{c(i)\geq j\}=\sum_{j=1}^{nk}P\{c(i)=j\} \tag{5}$$

也就是说,把公式(4)累加,直到 j 的可能最大值 nk.用阶乘的斯特林公式作了进一步推断,得到:

$$P\{c(i)\geq j\}\leq\binom{nk}{j}\frac{1}{m^{j}}\leq\left(\frac{enk}{jm}\right)^{j} \tag{6}$$

下面根据以上结果推断在 Filter_CBF 中的误判率.设阈值为 j,问题可以描述为共有 m 个计数器,对于其中的 k 个计数器,其中最小的值大于等于 j 的概率.由于 m 个计数器是相互独立的,且单个计数器大于等于 j 的概率是相等的,所以全部计数器大于等于 j 的概率等于单个计数器大于等于 j 的概率相乘,即等于单个计数器大于等于 j 的概率的 k 次方,也即公式(6)的 k 次方,得到:

$$P_{k}[c(i)\geq j]\leq\left(\frac{enk}{jm}\right)^{kj} \tag{7}$$

这个值就是所要求的 Filter_CBF 的误判率,即

$$P_{filter\_CBF}=\left(\frac{enk}{jm}\right)^{kj} \tag{8}$$

从公式(8)可以看出,在用于过滤长流的 Filter_cbf 中,影响误判率的因素有 4 个:已经被放入的报文个数 n、Hash 函数的个数 k、Hash 空间中计数器的大小 m 以及长流定义的分隔点 j.

总结公式(3)和公式(8)这两个误判率公式可以看出,影响误判率的主要是 4 个参数:

n:已经被标记的元素个数.根据使用场合的不同,在公式(3)中是被标识的长流个数,在公式(8)中是被标识的报文个数.这是一个反映处理过程的计数器。计数器的值可以增加或者减少,反映的是这个 CBF 数据结构的一个瞬时值,值的变化是无法人为控制的.但是,随着这个值的增加,误判率在增加,因此在达到一定的值之后,系统必须将 CBF 清空，以获得较小的误判率。

k:Hash 函数的个数。这个值是算法实现时必须确定的值,一般会在时间耗费和准确性之间权衡确定.有关 k 的取值讨论在文献[13]中有:当 $k=\ln 2m/n$ 时,可以得到最小的误判率.可见,k 的最优取值取决于计数器的个数和已经被标识的元素个数之比 m/n,k 小于或者大于这个最优值都会导致误判率高于最小误判率.但是,m/n 在算法进行的过程中随着 n 不断变化,因此只能取个相对较优的值;此外,因为 k 越大,需要计算的 Hash 函数个数越多,

在本文第 3.4 节中的时间测试表明,本算法的主要时间耗费是 Hash 函数的计算,因此,k 的选择以误判率在可接受范围内为宜.

m:CBF 结构中计数器的数目.这个值是影响误判率的关键因素,m 的值越大,误判率越小;但是内存资源的消耗是会受到限制的,因此必须确定这个值对算法准确性的影响程度,本文第 3.2 节将着重研究这一点.

j:这个参数存在于公式(8)中,对 Filter_CBF 的误判率产生影响,其物理含义为本算法在用于长流识别的时候所施加的长流定义值.这个值根据应用目的的不同而设定,随着 j 的增长,误判率呈非线性的变化,因此在相同的资源情况下,需要根据被采集的流数据的最终用途定义 j 的取值.

从误判率的公式来看,误判率不仅受到硬件资源的影响,也受到被处理数据特性的影响.不同的被处理数据重尾分布的不一样,会导致误判率的不同.因此,在使用该算法时,最小误判率未必能够达到,只需要保证误判率在可接受范围内即可.当误判率超出可接受范围时,采用清空计数器的方法就可以将误判率降低.

下面主要研究参数 m 的取值对算法准确性的影响,以确定在内存资源受到限制的情况下,本算法的实用性.

3.2 内存大小对CCBF准确性的影响

从误判率公式可以看出,Hash 空间的大小(计数器的多少)直接影响了算法的准确性.本文采用模拟数据验证算法中 CBF 使用的计数器数目对 CCBF 长流识别准确性的影响.

采用一个报文仿真系统来产生模拟数据.该系统使用 C 语言完成,采用蒙特卡罗方法,用随机变量仿真各维网络测度以产生近似于真实流量的报文.其中,随机变量的生成使用了 GNU Scientific Library 软件(版本 1.10).按照流到达服从泊松分布,流长分布服从 Pareto 分布的方式产生了测试数据流,被测试数据报文总数目为 2 239 407 个.

在实验中,我们不断调整 Hash 空间中计数器的数目,将得到的流长分布向量与仿真系统产生的流长分布向量进行比较.比较的方法是使用向量之间的欧式距离.

设仿真系统产生的流长分布为 $y_1,y_2,\ldots,y_n$.第 j 次识别得到的流长分布记为 $x_{j1},x_{j2},x_{j3},\ldots,x_{jn}$,则这两个流长分布向量之间的欧氏距离为

$$d_{Y,Xj_k}=\sqrt{\sum_{i=1}^{n}(y_i-x_{ji})^2} \tag{9}$$

欧氏距离越小,说明两个向量之间越接近.

图 3 是在不同 Hash 空间计数器数目的情况下得到的流长分布和测试数据的流长分布之间欧式距离的变化图.

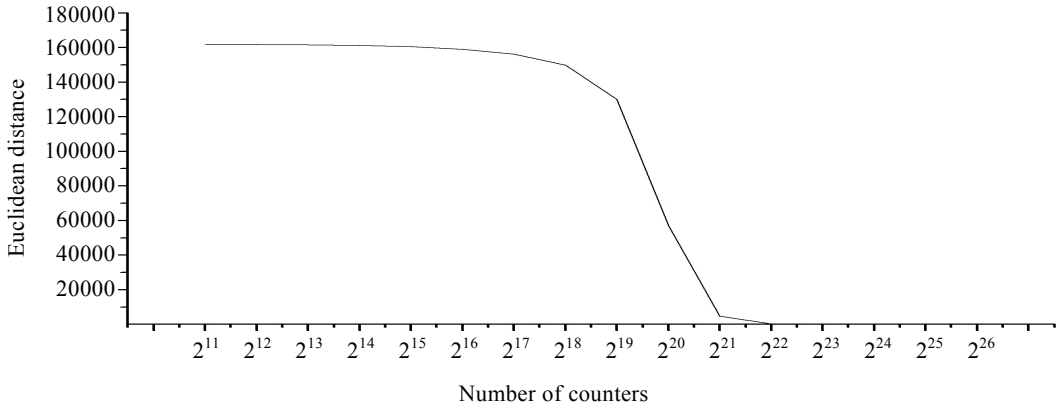


Fig.3 Effect of the Hash space on the Long flow identification error rate

图 3 Hash 空间对长流识别错误率的影响

从实验结果可以看出,在对这个模拟数据进行处理时,当 Hash 空间中的计数器达到 2^{22}(4.2Mb 内存)以后,欧氏距离为 0,也就是说,得到的流长分布和真实情况完全一样.这相当于将长流的阈值设置为 1,已经是一个极

端的情况,如果根据应用需求设置阈值,所需要的计数器个数则远远小于这个值.在目前的硬件成本情况下,可以认为本算法在对内存的需求并不大的情况下可以保证结果的正确性.

3.3 在相同内存限制条件下与其他长流识别算法的准确性比较

下面将 CCBF 算法的检测精度与相对来说精度最高的 MF 算法进行比较.

为了公平,与 CCBF 一样,将 MF 中每个计数器上进行字节累加改成报文数目累加.比较这两种算法在内存资源耗费相同时的错误率.对于长流检测算法来说,一般使用长流漏检率和平均误差两个指标来评价算法的性能.长流漏检率为算法检测到的长流与实际长流个数的比值,平均误差是所有长流的字节测量误差绝对值与所有长流总字节数的比值.根据本文对长流的定义,将平均误差中的字节数改成报文数得到平均误差.

由第 3.2 节可知,在内存足够大的情况下,可以保证检测到的长流数据完全正确.本文对已有的 trace 分别在内存足够大的情况下计算其流长分布,对每一组数据,在内存不断增大的情况下分别计算其流长分布.内存大到一定程度后,流长的分布不再有任何变化,认为此时得到的值是实际值.然后,在内存一致的情况下,不断增加需要检测的报文量,将得到的检测结果和实际值进行比较,得到漏检率和平均误差.

本文所选取的 TRACE 来源于:1) 美国互联网研究国家实验室(National Laboratory for Applied Network Research,简称 NLANR)公开提供的 TRACE;2) CERNET 华东北地区网络中心采集的 TRACE.Abilene 系列的 TRACE 来自于从 Indianapolis(IPLS)到 Cleveland(CLEV)主干网络,CERNET 采集点位于江苏省教育网边界路由到国家主干路由之间,采用华东北地区网络中心设计开发的高速网络采集系统——Watcher,使用分光的方式采集,各采集器之间采用 NTP(network time protocol)对时,其时间精度和采集丢包率均在可控的极小范围内[16].

在实验中,根据文献[14]中的研究结果,多个主干网 trace 在流长等于 10 的地方流分布出现拐点,设报文长度大于等于 10 的流为长流,CCBF 用于记录流存在和过滤的 Hash 空间都为 524 288 个计数器,占用内存空间为 800K.Hash 函数个数取 8,相应地,MF 算法中设过滤的 Hash 空间为 8 阶,一共占用的内存空间也为 800K,具体设置换算的原理参见文献[8].图 4 为使用 CCBF 和 MF 对 Abilene-I 和 CERNET 的两个主干网的 trace 进行处理,随着处理报文数的增长而变化的长流误判率变化图.图 5 为相应的报文平均误差.

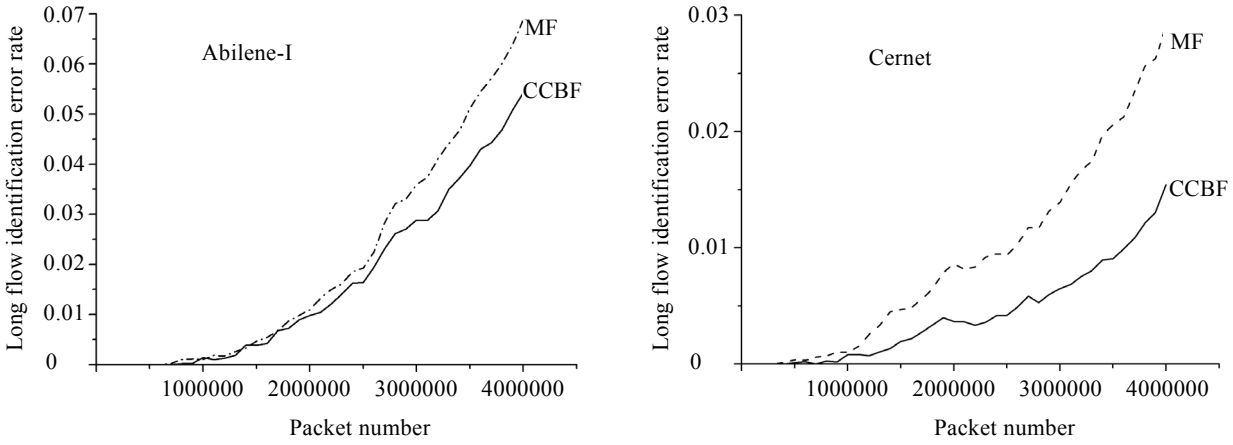


Fig.4 Long flow identification error rate with the increasing of the packet number

图 4 长流误判率随着报文数目增加的变化趋势

由图 4 和图 5 可以看出,在内存大小足够的情况下,CCBF 算法的错误率随着处理的报文数目增加逐步体现出其优于 MF 算法.

直观地看来,这两种算法都使用了 Hash 函数定位的计数器作为“filter”过滤长流.Hash 函数会把不同的输入值映射到同一个输出值,属于不同的流的报文可能被映射到 Hash 表中的同一个位置.这会导致两个或多个流的报文在同一位置进行累加,从而多个短流的报文数被累加后会超过阈值,被当成一个长流.另一方面,一些短流的计数被累加到长流的计数器上,造成长流数据的不准确.两种算法的区别在于:CCBF 将过滤器和长流识别分

成两个空间,在过滤空间中将识别出的长流对应的计数器值相应减少阈值的量.这样,被识别出的长流不会对后继流判别造成影响;而 MF 算法需要依赖长流的存在判断是否属于已知的长流,对已经被识别出的长流不能进行删除操作.这样,被判别出的长流越多,对后继长流过滤的影响越大.因此,在不清空 Hash 空间的情况下,随着处理报文数目的增加,CCBF 算法要优于 MF 算法.图 4 和图 5 使用报文数定义了长流.此外,我们采用 MF 算法中用字节数作为长流定义的方法重新做了上述实验,CCBF 的误判率仍然优于 MF.从原理上看,上述分析同样适用,因此可以认为,在两种长流定义下,CCBF 的误判率在报文数增多的情况下都是优于 MF 算法的.

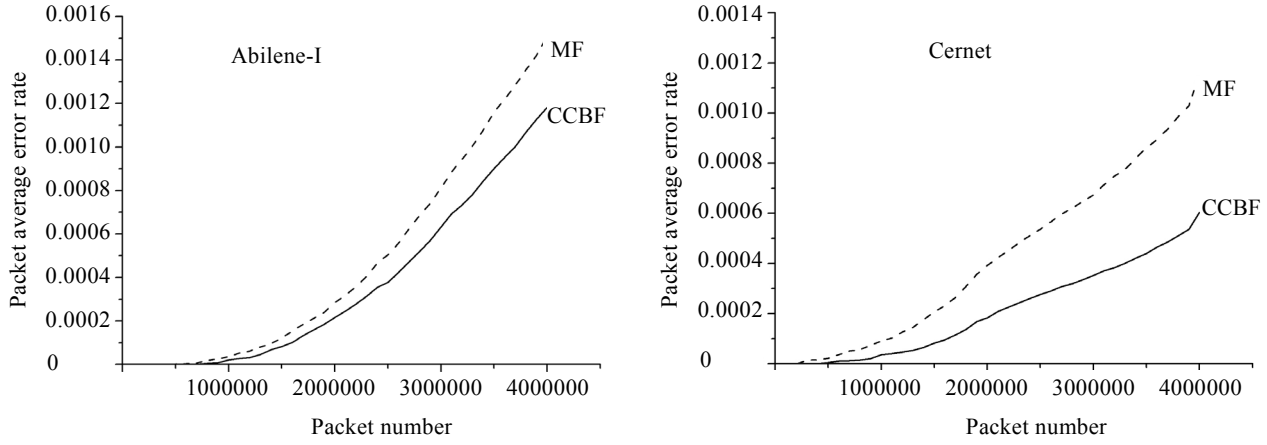


Fig.5 Packet average error rate with the increasing of the packet number

图 5 报文平均误差随着报文数目增加的变化趋势

由于网络流的重尾分布,短流的数目远远大于长流的数目,当需要识别的长流的阈值定义在较小的值时,会导致需要识别出来的流数目较多,MF 会有较大的误判率,CCBF 结构的优越性就比较明显.从图 4 和图 5 可以看出,在长流定义为报文数大于等于 10 的情况下,当处理报文数目达到 400 000 时,MF 的误判率明显高于 CCBF 算法.

3.4 CCBF算法的时间效率

对该算法的执行时间分析,需要看每个报文的处理时间.从算法对每个报文的处理流程来看,对一个报文的处理时间按照判断结果的不同有不同的过程.但是考虑到网络流量的重尾分布特征,绝大部分报文经过的都是最短的处理流程.下面对每报文的处理时间从实践的角度进行分析.

定义以下的时间耗费标识:

T_{hash}:使用 Hash 算法对一个报文的流标识计算出 k 个 Hash 值的时间.

$T_{judge\_longflow}$:在 longflow_CBF 中判断是否是已经存在的长流的时间.

$T_{update\_longflow}$:修改 longflow_CBF 的时间.

T_{judge_filter}:在 filter_CBF 中判断是否是识别出一个长流.

T_{update_filter}:根据计算得到的 Hash 值更新 CBF 中计数器的时间,包括对各个计数器加 1、减 1 都用这个标识.

根据各个报文的标识得到的判断结果,每报文处理流程是以下 3 种情况中的 1 种:

$$\begin{cases} t_1 = T_{hash} + T_{judge\_longflow} & \text{(属于长流)} \\ t_2 = T_{hash} + T_{judge\_longflow} + T_{judge\_filter} + T_{update\_filter} & \text{(属于短流和未知长流)} \\ t_3 = T_{hash} + T_{judge\_longflow} + T_{judge\_filter} + T_{update\_filter} + T_{update\_longflow} & \text{(属于长流,而且该长流首次被过滤出)} \end{cases} \tag{10}$$

假设属于这 3 种情况的概率分别为 p_1,p_2,p_3,这个平均时间是

$$\overline{t} = p_1 t_1 + p_2 t_2 + p_3 t_3 \tag{11}$$

将公式(10)代入公式(11),考虑到 $p_1+p_2+p_3=1$,得到:

$$\overline{t} = T_{hash} + T_{judge\_longflow} + (p_2 + p_3)(T_{judge\_filter} + T_{update\_filter}) + p_3 T_{update\_longflow} \tag{12}$$

在 Pentium(R) D CPU 3.40GHz 的 PC 上,排除硬盘读写带来的影响,通过随机产生报文的五元组进行多次测试,得到几个主要操作的平均耗费时间,见表 1.

Table 1 Average time of the main operations of CCBF algorithm (μs)

表 1 CCBF 算法的主要操作耗费的平均时间 (μs)

T_{hash}	$T_{judge\_longflow}$	T_{judge_filter}	$T_{update\_longflow}$	T_{update_filter}
5.349 5	0.122 66	0.123 9	0.125 7	0.146 1

从表 1 中 CCBF 算法的计算时间耗费可以看出,Hash 算法花费的时间远远大于其他操作的时间.在本算法实现中,使用的是 SHA 算法,该算法用软件实现时间耗费较大,但在系统应用中可以改用其他简单 Hash 算法或者用硬件芯片实现,都可以大大提高 T_{hash} 的速度.如果用硬件实现,使用本文中的 SHA 算法,速度至少可以提高 10 倍,也即 T_{hash}≤0.53495μs.将表 1 的结果带入公式(12)得到:

$$\bar{t} = 0.27p_2 + 0.3957p_3 + 0.6571 \tag{13}$$

为了计算每报文处理的平均时间,需要知道 p_2,p_3 的值.最现实和直观的方法是以样本频率代替概率,将样本中的报文分成前 3 种不同的报文,得到每种报文所占的比例就是频率.本文用 Abilene-I 和 Cernet 的两个 trace 考察报文的分布.

对这些 trace 使用 CCBF 算法进行长流识别发现,p_1,p_2,p_3 所占的比率与长流的定义有着极大的关系.使用本算法对 Abilene-I 和 Cernet 的 trace 进行分析,报文数都是 100 000 000 个, Abilene-I 的时间跨度是 569s,Cernet 的时间跨度是 367s.图 6 是在不同长流阈值定义的情况下,经过 3 个主要处理分支的报文比率变化图.

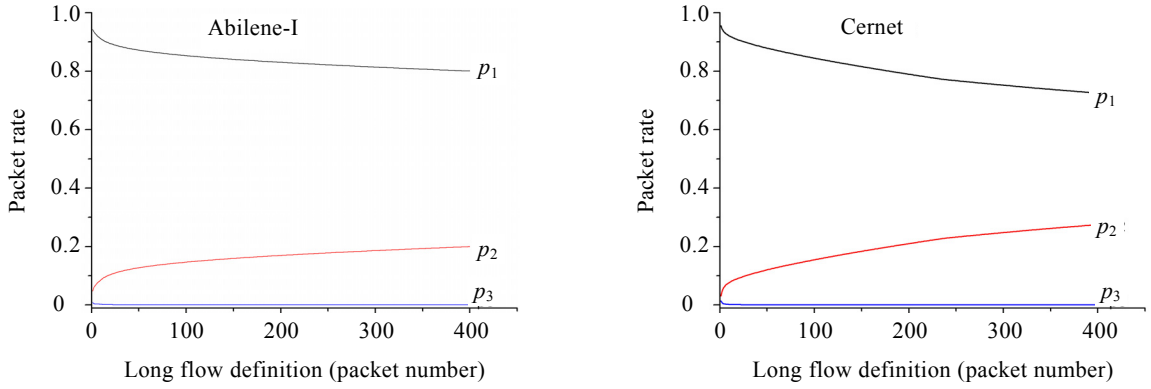


Fig.6 Packet rate variation in the main process branches

图 6 主要处理分支报文比率变化

从图 6 中可以看到,p_1 所占比率随着长流定义的增加而下降,但是下降的趋势很平缓,也就是说,大部分报文都是通过最短的处理流程完成的.在应用本算法进行长流识别时,长流的定义值是根据应用需求而定义的一个常数值.通常,TCP 流的平均长度为 40 个报文左右.假设长流的定义为 40,将图中比率带入公式(13),这两个 trace 分析中的每报文平均处理时间分别为

$$\bar{t}_{\text{Cernet}} = 0.6878\mu s,$$

$$\bar{t}_{\text{Abilene-I}} = 0.6902\mu s.$$

由于各个 trace 中流长分布略有不同,得到的每个 trace 的每报文处理时间略有不同.但是对符合大规模主干网流长分布模型的各个 trace 进行计算,得到的结果则相差很微小.可以认为每报文处理时间是 0.69μs,按照这个每报文平均处理速度,该系统每秒可以处理 1 449 275 个报文,也就是说,可以处理最高 1 449kpps 的主干网的流记录识别.当然,这一结论是建立在两个基本的条件下:(1) Hash 算法用硬件实现;(2) 排除输入、输出的瓶颈,在测试运行时间的实验中用的数据是在内存中随机生成的.因此,算法本身虽然达到了非常理想的处理速度,但在实现中还需要排除输入、输出带来的瓶颈.考虑到测试使用的硬件平台是很普通的 PC,而且随机产生的数据没

有重尾分布的特点,耗时相对更多,在提高资源配置的情况下,CCBF 的处理速度完全可以达到 1 500kpps 主干网上流量识别的监测需求.

4 结束语

本文提出了一种基于双层 Bloom Filter 的长流识别算法 CCBF,分析了该算法的误判率,通过模拟数据研究了内存大小对 CCBF 准确性的影响,并在相同内存资源限制下的条件下,将该算法与类似算法的准确性进行了比较.结果表明,由于该算法将长流过滤和长流存在分开标识,在数据量较大的情况下,该算法具有较小的平均错误率.对该算法的时间效率分析表明,在排除输入、输出影响的情况下,该算法的处理速度可满足 1 500kpps 主干网的检测要求,各项指标反映该算法可以应用于大规模主干网的流量采集.

References:

[1] Cisco Systems Inc. Cisco Netflow. http://www.cisco.com/en/US/products/ps6601/products_ios_protocol_group_home.html

[2] Claffy KC, Polyzos GC, Braun HW. Application of sampling methodologies to network traffic characterization. Computer Communications Review, 1993,23(4):194−203. [doi: 10.1145/167954.166256]

[3] The Internet Engineering Task Force. IETF packet sampling work group. http://www.ietf.org/html.charters/psamp-charter.html

[4] Estan C, Varghese G, Fish M. Bitmap algorithms for counting active flows on high speed links. IEEE/ACM Trans. on Networking, 2006,14(5):925−937.

[5] Humar A, Sung M, Xu J, Wang J. Data streaming algorithms for efficient and accurate estimation of flow size distribution. Performance Evaluation Review, 2004,32(1):177−188. [doi: 10.1145/1012888.1005709]

[6] Zhou MZ. Study of large-scale network IP flows behavior characteristics and measurement algorithms [Ph.D. Thesis]. Nanjing: Southeast University, 2006 (in Chinese with English abstract).

[7] Kim MS, Won YJ, Hong WJ. Characteristic analysis of Internet traffic from the perspective of flows. Computer Communications, 2006,29(10):1639−1652. [doi: 10.1016/j.comcom.2005.07.015]

[8] Estan C, Varghese G. New directions in traffic measurement and accounting: Focusing on elephants, ignoring the mice. ACM Trans. on Computer Systems, 2003,21(3):270−313. [doi: 10.1145/859716.859719]

[9] Kodialam M, Lakshman TV, Mohanty S. Runs based traffic estimator (rate): A simple, memory efficient scheme for per-flow rate estimation. In: Zhang ZS, Low S, eds. Proc. of the IEEE INFOCOM. New York: IEEE Communications Society, 2004. 1808−1818.

[10] Duffield N, Lund C, Thorup M. Charging from sampled network usage. In: Paxson V, ed. Proc. of the ACM SIGCOMM Workshop Internet Measurement. New York: ACM Press, 2001. 245−256.

[11] Wang HB, Pei YJ, Lin Y, Chen SD, Jin YH. A LRU based algorithm for identifying and measuring large flow. Journal of Electronics & Information Technology, 2007,29(10):2487−2492 (in Chinese with English Abstract).

[12] Bloom BH. Space/Time trade-offs in hash coding with allowable errors. Communications of the ACM, 1970,13(7):422−426. [doi: 10.1145/362686.362692]

[13] Fan L, Cao P, Almeida J, Broder AZ. Summary cache: A scalable wide-area Web cache sharing protocol. IEEE/ACM Trans. on Networking, 2000,8(3):281−293. [doi: 10.1109/90.851975]

[14] Zhou MZ, Gong J, Ding W, Cheng G, Long flows' information statistics based on MGCBF algorithm. Journal of Southeast University (Natural Science), 2006,36(3):472−476 (in Chinese with English abstract).

[15] Border A, Mitzenmacher M. Network applications of bloom filters: A survey. Internet Mathematics, 2005,1(4):485−509.

[16] Shi B, Ding W, Gao YD, Gong J. IP trace data based on a CERNET backbone. Journal on Communications, 2006,27(11A): 214−217 (in Chinese with English abstract).

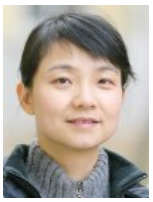
附中文参考文献:

[6] 周明中.大规模网络 IP 流行为特征及其测量算法研究[博士学位论文].南京:东南大学,2006.

[11] 王洪波,裴育杰,林宇,程时端,金跃辉.基于 LRU 的大流检测算法.电子与信息学报,2007,29(10):2487−2492.

[14] 周明中,龚俭,丁伟,程光.基于 MGCBF 算法的长流信息统计.东南大学学报(自然科学版),2006,36(3):472−476.

[16] 史冰,丁伟,高亚东,龚俭.一个基于 CERNET 主干信道的 IP 流数据 TRACE.通信学报,2006,27(11A):214–217.



吴桦(1973－),女,江苏南京人,博士生,讲师,主要研究领域为网络管理,网络行为.

龚俭(1957－),男,博士,教授,博士生导师,主要研究领域为网络安全,网络行为,网络体系结构.

杨望(1979－),男,博士,讲师,主要研究领域为网络安全,网络管理.