基于 TCBF_LRU 的高速网络大流检测算法

白磊 田立勤 陈超
华北科技学院计算机学院，北京东燕郊 101601
(leib@ncist.edu.cn)

A TCBF_LRU Algorithm for Identifying and Measuring Elephant Flows in High Speed Network Flows

Bai Lei, Tian Liqin Chen Chao
(College of Computer Science, North China Institute of Science and Technology, Yanjiao 101601)

Abstract In the high-speed backbone network, with the increasing speed of network link, the number of network flows increase rapidly. Meanwhile, with restrictions on hardware computing and storage resources, so, how to identify and measure elephant flows timely and accurately in massive data become a hot issue in high speed network flow measurement area. In this paper, we propose a new algorithm based on TCBF_LRU to realize elephant flow identification, according to the defect of traditional LRU algorithm which discards elephant flows easily and update system frequently when large numbers of mice flows arrive. By using time out and pre-protection method, the algorithm can filtrate most of the mice flows, reduce the probability of mice flows displacement elephant flows in LRU algorithm, and improve the accuracy of the algorithm. The complexity and error rate of the algorithm was analyzed. The influence of elephant flow measurement accuracy for parameter configuration was analyzed through the actual backbone trace data. The theoretical analysis and the simulation result indicate that compare to the standard LRU algorithm and BF_LRU algorithm, with the same cache space, our algorithm can identify elephant flow more accurately and practicality.

Key words network measurement ; massive data; network flow; hash method; least recent used

摘要 在高速主干网络中，随着网络链路速率的不断提高和网络流数量的急速增加，同时受到硬件计算和存储资源的限制，如何及时、准确地在海量数据中，有效的检测出其中的大流信息，成为目前大规模高速网络流测量的热点问题。本文根据传统的 LRU 算法容易淘汰大流和频繁更新给系统带来巨大压力的缺陷，提出一种新的大流检测算法——TCBF_LRU 算法，用于高速网络大流信息识别。算法通过时间超时和大流报文预保护策略，过滤大部分的小流报文，极大减少 LRU 算法小流置换大流的概率，提高算法的准确性。分析了算法的误判率和复杂度，并通过实际主干网 trace 数据，实验分析了算法参数配置对于大流检测准确性的影响。理论分析和仿真结果表明，与标准 LRU 算法和 BF_LRU 算法相比，在使用相同的缓存空间下，TCBF_LRU 算法具有更高的测量准确性和实用性。

关键词 网络测量；海量数据；网络流；哈希；LRU

中图法分类号 TP393

近年来，随着互联网的快速发展和网络新应用的 不断出现，计算机网络呈现向高速化、大规模、复杂

收稿日期：2014-11-10
基金项目：973 计划专项(No.2011CB311809)；国家自然科学基金（No.61163050）；中央高校基本科研业务费（No. 3142014085, 3142014100）

化方向发展的趋势，显著特点是产生的数据量大、数据分组到达频率高，导致单位数据分组的处理时间越来越短，对系统的存储能力、处理能力和传输能力都提出了极大的挑战，这便对网络流量处理设备提出更高的要求。目前在高速主干网络上，处理每个分组的平均时间必须在纳秒级完成。例如，在 OC-192 链路上，处理一个数据分组平均时间约为 32ns。而在 OC-768 链路上，分组平均处理时间为 8ns[1]。并且随着主干网链路带宽的增加，对实时性的要求会更高。因此，针对海量数据，如何及时、准确的挖掘出有用的信息，成为当前高速网络研究的热点。而基于网络流(flow)的测量方法为高速网络流量监控开辟了新的途径，通过把数据包归并到相应流中，极大地压缩了数据量，使得网络数据的存储、处理和传输更为容易。

目前很多基于网络流测量的研究表明，即便在不同的网络环境中，网络流的统计呈现很强的重尾分布特性[2,3]。这种特性被称为“the elephants and mice phenomenon”，即大部分流（mice 流）只产生很少数量的报文，而小部分流（elephant 流）却产生很大数量的报文。大流的一个很显著的特性就是它们仅占总体流的数目的一小部分却产生总流量的绝大部分。这样，在实际应用中，很多情况下只需掌握占据大部分流量的大流信息即可满足需要，如网络流量记账等。因此，利用有限的硬件资源关注大流，尽可能收集字节流量超过某个固定阈值的大流成为一个较好的选择。

Cristian Estan 等人首次采用“抓大放小”策略将大流检测引入流量测量领域，并提出抽样保持(sample and hold）和多级过滤（multistage Bloom filters）算法用于长流识别，前者算法简单、易于实现，但当网络流数量较多，而抽样频率较低时需维护较大的存储空间，且误差偏高。后者对存储空间需求较小，处理速度较快，但算法的错误肯定率较高，会把小流误检为大流[4]。Tatsuya Mori 提出一种使用周期抽样识别长流的机制。这一机制的关键是在恰当地权衡错误肯定和错误否定的基础上，使用贝叶斯理论计算出抽样流的报文数的阈值，通过这一阈值可以判定其原始流是否是长流。但它的缺点是实现时精度不高，错误的肯定率和错误的否定率之间的平衡不易实现。Duffield 提出由抽样流数据推断出原始流数据、获取原始流信息的思想，并提出两种推断方法：比例法和 EM 算法[5]。程光提出使用积分推断法和迭代法根据未抽样流数和已抽样流数推断原始流量的统计信息[6]。但是使用报文抽样技术，会造成一些内在信息的损失，所以往往使用推断的方法来减少这种损失，但是估计方法像 EM 算法计算量太大，不适合数据的在线处理。Smitha 提出在路由器队列管理中，利用页面置换策略 LRU（least recent used）机制来识别持续时间长、高速率的 IP 流[7]。张震等提出将流识别和流过滤分开处理的方式使用 Bloom Filter 及 LRU 算法实现长流信息统计[8]。使用 LRU 的方法具有处理速度快、识别率高、存储空间可控等优点，但在实际测量中，当流的数量较多，某些突发性的小流有可能导致大流量对象被替换出缓存[9]，从而引起较大的测量误差。

本文根据网络流量分布的特性，提出基于 TCBF_LRU 的大流检测算法，算法能够在使用较小的高速缓存资源情况下，精确提取大流量对象。

1 网络流的定义

网络中的流概念可以定义为对一个呼叫或连接的人为逻辑对应。流是流量的一部分，由起始时间和停止时间定界。与流相关的属性值（源/目的地址、分组计数、字节计数等）具有聚合性质，反映了在起始和停止范围发生的事件。由于研究背景的不同，对于流采用了不同的定义。

定义 1　流是指符合特定的流规范和超时约束的一系列数据包的集合。在本文中流是指相同的源 IP，宿 IP，源端口、宿端口的按超时约束的 TCP 或 UDP 报文集合。流超时决定什么时候结束一个流，即同属一个流的相邻两个报文的到达时间间隔，在本文中流超时都设置为 30 秒。

定义 2　大流是一段时间内某个流的长度超过事先定义的阈值 *TH* 的流。

定义 3　高速网络大流监测算法的约束条件：1）存储空间有限。用来维护摘要统计信息且所需存储空间远小于网络流空间的大小，通常只存储少量数据项的摘要信息；2）实时处理。对每个数据项的处理时间很短，操作少而简单。3）一次线性扫描。每个报文的到达次序完全随机，只能依序从头至尾读取一次数据流[10]。

2 标准的 LRU 算法

LRU 算法又称为近期最少使用算法，其基本原理是：维护固定大小的缓存空间，将到达的元素依次存储到缓存中，并始终保持最新到达的元素置于缓存的顶端，而最久未到达的元素则保留在缓存的最底部。当有新元素加入而缓存已满情况下，把缓存最底部的元素替换出去，并将新元素置于顶部。LRU 算法在

计算系统中有着广泛的应用，如内存管理、数据库缓存管理以及磁盘缓存管理等领域。

文[7]首先将其引入网络流量测量领域，通过维护一个固定大小的 LRU 高速流缓存，并使用 LRU 思想对到达的每个分组所属的流标识进行置换。由于小流持续时间短、长度小、到达速率低，根据 LRU 算法总有可能被替换出去；而大流由于持续时间长、长度大、访问缓存频繁，所以往往会留在 LRU 缓存的顶部，从而可以实现大流检测。

然而在大规模网络链路测量过程中使用传统的 LRU 算法还存在一定的限制。一方面，由于主干网流量的巨大性和突发性，而且由于网络重尾分布的特点，网络中流信息大部分为小流信息，因此在极短的时间内，流标识会填满整个 LRU 缓存空间，从而导致大流信息被置换出 LRU 缓存空间。文献[7]的实验结果也表明有 10%至 20%的大流信息没有检测到。另一方面，由于受硬件资源的限制，LRU 缓存空间也有限，这样随着测量时间的延长，流数量的增加，LRU 缓存频繁更新，不仅会丢失前阶段测量的信息，导致测量精度降低，而且也给系统硬件设备带来很大的压力。因此，如何能够将网络流量中的大部分的小流信息过滤掉，减少大部分的小流信息进入 LRU 缓存的概率，成为解决 LRU 算法测量准确性和实用性的关键问题，本文提出基于 TCBF_LRU 的大流检测算法，可以有效减少小流进入 LRU 缓存的概率，减少 LRU 更新的频率，有效实现实时的大流信息识别。

3 TCBF_LRU 算法

3.1 标准 Bloom Filter

标准的 Bloom Filter(BF)是用来表示一个集合的随机数据结构，它支持成员查询、随机存储[11]。其工作原理是：对于一个源串集合 $S=\{x_1, x_2 ..... x_n\}$，BF 申请一个内存大小为 M 单元的存储空间 A，每个单元维护一个计数器初始值置 0，并使用哈希函数集合 $H=\{H_1, H_2, ...H_k\}$，每一个 H_i，均是一个哈希函数。对于源串集合 S 中的任何一个元素 x_i，通过集合 H 中的 k 个独立的哈希函数映射到存储空间 A 中，得到 k 个[1..M]之间的数，并将内存空间 A 中的这 k 个对应单元比特位置 1。因此 Bloom Filter 随机存储的过程就是通过 k 个哈希函数映射到 Bloom Filter 的存储空间，并把相应的位置置 1 的过程。当 BF 需要判断任一元素 x' 是否属于集合 S 中的元素时，BF 对 x' 经 k 个哈希函数作用判断哈希位置是否均为 1，如果是则认为 x' 属于集合 S，否则就不是集合中的元素。然而当对 x' 哈希的结果均为 1，而实际 x' 却不属于集合 S 时，就出现了错误的肯定。Bloom Filter 错误肯定的 x' 属于集合 S 的概率，即为错误肯定率（False Positive Rate，FPR）。研究表明，Bloom Filter 长度的下界，以及所用的哈希函数的个数，与其错误肯定率和错误否定率之间存在定性的关系。BF 存储和查找过程如图 1 所示。

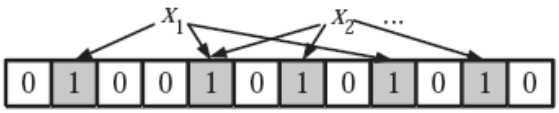

图 1 标准 Bloom Filter 原理示意图

3.2 Time Bloom Filter 过滤小流报文

Time Bloom Filter 是由 BF 改进而来，与 BF 不同的是 TBF 中的 m 个单位维护的是报文的时间戳而不是 0 或 1。当一个报文到达时，使用 k 个哈希函数对报文标识进行运算并哈希到相应 m 个单元，判断 k 个对应单元两个时间戳的差值是否大于某个预先设定的超时时间 t_0，如果有任何某个单元的时间差大于 t_0，则丢弃此报文。无论是否丢弃报文，每次 TBF 都将更新到达报文哈希空间的时间戳，这样 TBF 更新操作仅更新对应的时间戳，而不需要频繁重置。研究表明，流长度越长，其所属报文出现频率越高，报文的平均报文到达时间间隔越小[12]。由于同属一个流的报文，会哈希到同一空间。所以，对于同属某个大流的报文项，由于时间间隔小于阈值 t_0，只要在两个报文的时间间隔内，该哈希空间没有被其他流占用，这部分报文将不被丢弃。而对于大量的小流报文，由于不同流的报文被哈希到不同的空间，必然会导致某个单元的时间戳差值大于时间间隔 t_0，从而可以实现过滤大部分的小流报文。但极端情况下，当映射到大流的哈希空间频繁的被其他流占用时，TBF 也会丢弃大量的同属某个大流的报文，从而降低统计的准确性，因此需要使用另外一种过滤技术 CBF，减少丢弃大流报文的数量。

3.3 Counting Bloom Filter 预保护大流报文

Counting Bloom Filter 也称为计数型 BF，是由 Fan 等人提出的。与 BF 不同的是 CBF 中的 m 个单位维护的是一个计数器。这样 CBF 不但具有 BF 的添加和查询操作，也具有删除操作。添加操作是把存储空间的这 k 个对应单元的计数器加 1。删除操作是把存储空间的这 k 个对应单元的计数器减去某个值。由于同属相同流的报文，在 CBF 哈希位置相同，这样所属

大流的报文，即便超时，也可以提取到。因此可以使用 CBF 结构减少 TBF 中丢弃大量的大流的报文，起到大流报文预保护的作用。

3.4 使用 TCBF_LRU 算法检测大流

TCBF_LRU 算法由 TCBF 过滤模块和 LRU 识别模块两部分组成，其中 TCBF 模块又分为 TBF 和 CBF 两部分，基本结构如图 2 所示。

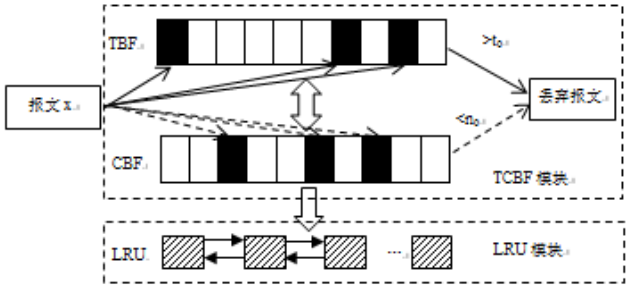

图 2 TCBF_LRU 算法过程原理图

TCBF_LRU 算法分别为 TCBF 模块中的 TBF 和 CBF 结构各分配一个内存大小为 M 单元的存储空间 $A1$ 和 $A2$，为 LRU 模块分配存储空间 $A3$。TBF 模块的每个单元维护报文到达的时间戳，而 CBF 模块的每个单元维护一个计数器，初始值置 0。TBF 和 CBF 分别使用哈希函数集合 $H=\{H_1,H_2,....H_k\}$ 中的不同的哈希函数对到达的报文的流标示（协议类型、源/宿 IP、源/宿端口）x_i 进行作用，分别通过集合 H 中的 k 个独立的哈希函数映射到存储空间 $A1$ 和 $A2$ 中，得到 k 个[1..M]之间的数。LRU 采用散列双向链表的数据结构[13]，每个链表节点都存储流标识和流的报文计数器。

图 3 给出了 TCBF_LRU 算法的基本过程：当一个报文到达时，首先使用 TBF 进行哈希映射，判断 k 个对应单元两个时间戳的差值是否大于某个预先设定的超时时间 t_0，如果有任何某个单元的时间差大于 t_0，并且 x_i 在 CBF 中的哈希映射位置小于 n_0 则丢弃（过滤）此报文，否则将由 CBF 更新。无论时间间隔是否大于 t_0 都将更新 TBF 中的 k 个单元的时间戳，以保证 TBF 中保存的是最新到达的报文的时间戳。CBF 更新操作是将报文哈希的 k 个单元的计数器加 1，如果 k 个单元的计数器均大于等于预先设定的阈值 n_0，则不再累加。通过 TCBF 模块过滤以后，提取的报文或者为时间未超时小于 t_0 的报文，或者为时间已超时但哈希位置计数大于 n_0 的报文，这部分报文再转交 LRU 算法处理。LRU 算法先判断该报文所属流标识是否已在链表中，如果已存在，则对应计数器加 1，同时将该节点置于 LRU 链表顶部；当所属流标识不在链表中，需要创建新的流量记录时，若 LRU 链表已满，则根据“近期最少使用”原则，淘汰 LRU 链表的最后一个流；同时，为了降低大流漏检率，在淘汰流时，需要判断其是不是已经为大流。最终输出流长度大于指定阈值的流即为需识别的大流信息。

```
x=Packet Label;
initialize (TBF,CBF,LRU)
//初始化 TBF、CBF 和 LRU
compute(h(x),g(x))
//分别计算 TBF 和 CBF 哈希函数值
if (all location of TBF[hi(x)]i=1..k <t0)||(all
location of CBF[hi(x)]i=1..k>=n0) {
//如果对于 k 个哈希映射，TBF 哈希位置均超时，或者 CBF
所有位置大于阈值 n0
      add(min(CBF[hi(x)]i=1..k));
      //保守更新 CBF，最小值+1
      LRU();  //转调 LRU 算法
} else       //TBF 超时并且 CBF 哈希位置小于 n0
      discard(x); //丢弃报文
update timestamp (TBF[hi(x)]i=1..k);
//无论是否丢弃均需更新 TBF 对应哈希位置时间戳值
LRU(){    //LRU 算法
      if(Find(x) or not full){
     //如果已在 LRU 链表中，或链表未满时
            update(count); //流报文计数+1
            setTop();  //将该流标识节点置顶
     }
     else   //不在链表中，且链表已满
     eliminate(last); //淘汰最后一个流节点
}
```

图 3 TCBF_LRU 算法

从以上分析可以看出，TCBF_LRU 算法只在 TBF 大于 t_0，并且 CBF 计数小于 n_0 情况下丢弃报文。由于占网络流数量大部分的小流报文，被哈希到不同的空间，必然会导致某个单元的时间戳差值大于时间间隔 t_0，或者对应的 CBF 的计数< n_0，因此通过 TCBF 可以过滤掉大部分的小流信息，减少小流进入 LRU 的概率，从而提高 LRU 算法的准确性。

3.5 算法分析

TCBF_LRU 算法的错误概率主要包括两部分：TCBF 的错误肯定率，即把小流错误肯定为大流的概率；LRU 的漏判概率，即 LRU 淘汰大流的概率。

TCBF 的误判率由 TBF 和 CBF 共同产生。对于集

合 S 中的 n 个元素，TBF 哈希位置为空的概率为 $p^{'}=(1-1/m))^{k}\approx e^{-k/m}$，令 $p=e^{-k/m}$，则在某个位置出现错误肯定率为 $P_{TBF}=(1-p^{'})^{k}\approx(1-p)^{k}=(1-e^{-k/m})^{k}$。对于 CBF 的 m 个计数器，被增加 n_0 次的概率为 $P\{c(i)=n_0\}=\binom{nk}{n_0}\left(\frac{1}{m}\right)^{n_0}\left(1-\frac{1}{m}\right)^{nk-n_0}$，则大于等于 n_0 的概率为

$$P_{CBF}=\sum_{n_0=1}^{nk}\binom{nk}{n_0}\left(\frac{1}{m}\right)^{n_0}\left(1-\frac{1}{m}\right)^{nk-n_0}\le\binom{nk}{n_0}\frac{1}{m^{n_0}}\le\left(\frac{enk}{n_0m}\right)^{n_0}$$

由于 TCBF 抽取报文的条件是既满足 TBF 也同时要满足 CBF 条件，所以最终 TCBF 的误判率为

$$P_{TCBF}=P_{TBF}*P_{CBF}=(1-e^{-k/m})^{k}\left(\frac{enk}{n_0m}\right)^{n_0}$$

对于 LRU 漏判概率，由于在任意测量时间段内，流量大小服从位置参数为 1、形状参数为 α 的帕累托分布。假设在测量时间段内报文总数为 M，LRU 平均每隔 N 个报文建立一个新流标识，并淘汰链表最底部的某个流。建设某大流 F 大小恰好等于阈值 TH，则在连续 N 个报文中没有出现大流 F 的概率服从超几何分布：$\binom{TH}{0}\binom{M-TH}{N}/\binom{M}{N}$，当 $M\gg N$ 时，超几何分布可以近似为 $\left(1-\frac{TH}{M}\right)^{N}$ 因此大流 F 被淘汰的概率为：

$$P_{LRU}(F=TH)=P(F=TH)\left(1-\frac{TH}{M}\right)^{N}$$

将 $P(F=TH)=\theta/TH^{\alpha+1}$ 代入公式可得

$$P_{LRU}(F=TH)=\frac{\theta}{TH^{\alpha+1}}\left(1-\frac{TH}{M}\right)^{N}$$

其中 θ 为归一化参数 $\theta=\left(\sum_{i=1}^{M}i^{-\alpha-1}\right)^{-1}$

由于算法使用哈希函数集进行作用，TBF 和 CBF 对哈希空间的一次访问内存开销均为 $O(k)$，每次更新 TBF 的 k 个时间戳开销为 $O(k)$，更新 CBF 的开销为 $O(k)$。由于 BF 使用并行散列运算，算法实际执行时间接近 1 次散列运算所需时间。同时，LRU 链表采用散列双向链表，使用拉链法解决散列冲突，则平均查找长度为 $O(1+\beta/2)$，其中 β 为装填因子。

4 实例分析

为了验证上述本文提出的 TCBF_LRU 算法的有效性，本文使用采集于 NLANR 的 Trace 进行仿真试验，Traces 总共报文数 6187376 个，共有 68367 个流。TCBF_LRU 算法的 TCBF 模块分配固定存储空间大小 m= 2^{16}=65536，即哈希空间为[0..65535]。实验测量网络报文数据的原始分布如图 4 所示，从图中可以看出网络流的分布统计呈现重尾分布特性。

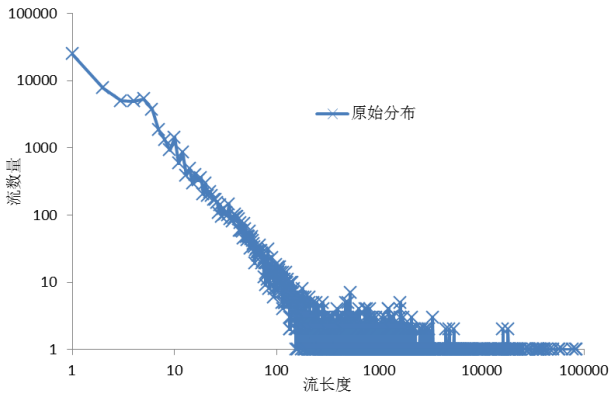

图 4 实验数据网络流原始分布图

根据文献[14]分析，基于异或、位移原则的比特流哈希算法在执行效率和哈希值的均匀性方面均高于 IPSX、CRC32 和 BOB 哈希函数，同时根据文献[3]和文献[8]可知哈希函数个数的最优取值取决于计数器的个数和已识别的元素个数之比，当 $k=\ln 2m/n$ 时取最小值。但由于随着算法测量过程的持续，n 的值不断变化，因此只能取相对最优值，以误判率在可接受范围内为宜，因此本文使用 XOR_SHIFT 系列哈希函数[14]，并且哈希函数个数为 $k=6$。

因此 TCBF_LRU 算法统计流信息需要配置的参数主要有：TBF 超时时间间隔 t_0、CBF 过滤阈值 n_0、大流阈值 TH 和 LRU 缓存空间 L。以 e_r 作为定量衡量 TCBF_LRU 算法的流识别统计的误差，即 $e_r=\frac{|N_f-N_f^{'}|}{N_f}\times100\%$，其中 N_f 表示实际大流个数，$N_f^{'}$ 表示实验识别统计的大流个数。

图 5 显示 TBF 丢弃报文比率与超时时间间隔阈值 t_0 变化时的关系。从图中可以看出，当 t_0=0 时，由于超时，且 CBF 计数值永远小于 n_0，所有的报文将都被丢弃，当 t_0 足够大时，绝大部分都不被丢弃，所以阈值 t_0 的取值可以确定过滤的报文数量，符合对 TBF 的过程分析。

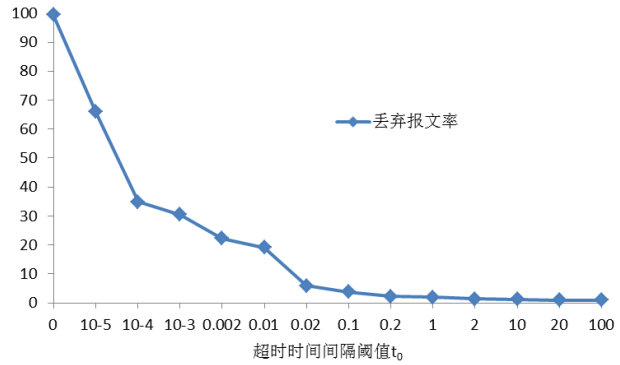

图 5 TBF 报文丢弃率与超时时间间隔阈值 t_0 的关系图

图 6 显示算法中当 CBF 过滤阈值 n_0 取固定值时，TBF 超时时间间隔 t_0 变化对不同的阈值的大流测量准确性的影响。从图中可以看出，当 t_0 小于阈值 0.1 秒时，测量误差，随 t_0 增加而减小，这是由于当 t_0 取较小值时，大部分的报文被丢弃掉，随着 t_0 增加丢弃报文减少，从而提高测量准确性。但当 t_0 大于阈值 0.1 秒时，测量误差，随 t_0 增加而增加，这是由于当 t_0 取较大值时，丢弃报文逐渐减少，大部分的小流没有被过滤掉，从而增加算法的错误肯定率，导致误差增大，而且算法对相对较小的大流的影响比较大的大流的影响要明显。考虑到算法的适应性，在实际测量时 t_0 取值[0.1,0.5]。

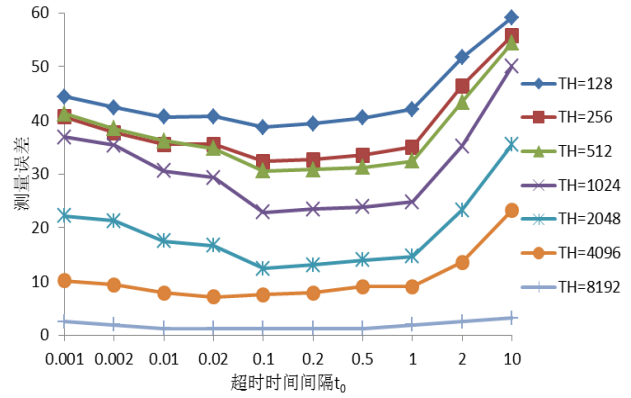

图 6 超时时间间隔 t_0 对大流阈值 TH 测量准确性影响图

图 7 显示算法中当 TBF 超时时间间隔 t_0 取固定值时，CBF 过滤阈值 n_0 变化对测量准确性和丢弃报文比率的影响。从图中可以看出，随着 n_0 值的增加，丢弃报文比率线性增加，这时由于当 n_0 增加时，长度小于 n_0 的流的报文都被过滤掉，但当 n_0 增加到一定值后，测量误差将逐渐增大，这是由于当 n_0 取较大值时，大流的部分报文也同时被丢弃，导致误差增大。考虑到算法使用的空间随 n_0 的增大而增大，n_0 取值[16,64]。

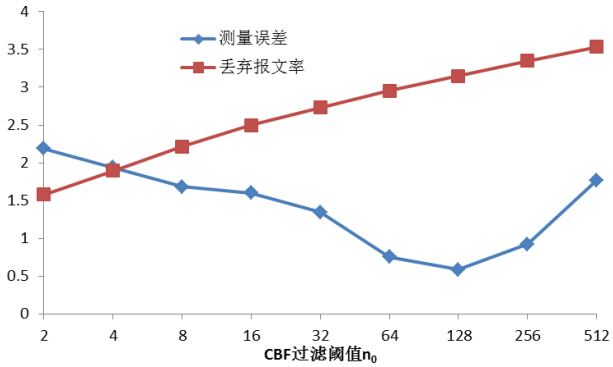

图 7 过滤阈值 n_0 对大流频繁项测量准确性和丢弃报文比率的影响图

图 8 和图 9 分别显示 TCBF_LRU 算法当参数 t_0=0.2 秒，n_0=16 时，与传统 LRU 算法在 LRU 缓存空间变化时测量不同阈值的大流的准确性比较。

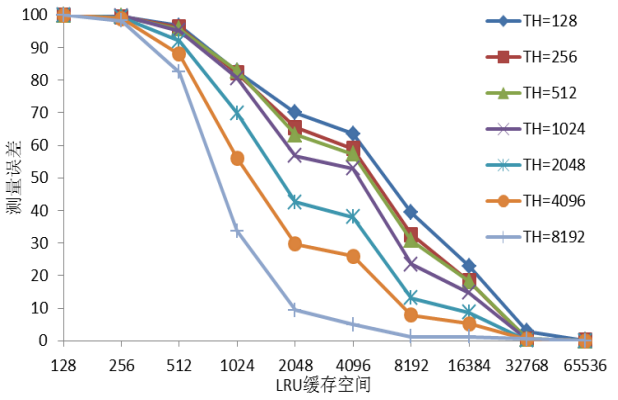

图 8 TCBF_LRU 算法 LRU 缓存变化时测量准确性图

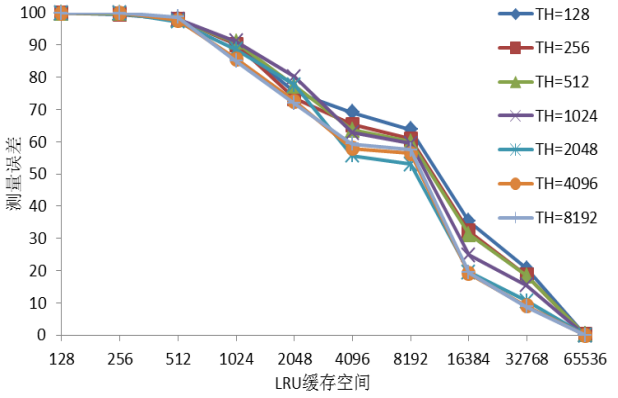

图 9 标准 LRU 算法 LRU 缓存变化时测量准确性图

从图中可以看出，当两种算法的 LRU 缓存空间较小时(128 时)，由于流数量巨大，而 LRU 缓存空间太小，导致绝大部分的大流信息被淘汰，测量准确性较差。当 LRU 缓存增加时，测量的准确性也随之提高，当 LRU 缓存足够大时，总是能将大流完全检测出来。但本文提出的 TCBF_LRU 算法的收敛速度远快于传

统的 LRU 算法，在使用相同的缓存空间条件下，测量的各种长度阈值的大流的准确性均比传统 LRU 算法准确。

图 10 显示以测量长流阈值 *TH*=8192 为例，TCBF_LRU 算法与文献[8]提出的 BF_LRU 算法及标准 LRU 算法，在使用相同的哈希函数和个数以及 LRU 缓存空间情况下测量准确性的比较。实验仿真结果表明，相对标准 LRU 算法和 BF_LRU 算法，TCBF_LRU 算法具有较高的测量准确性，尤其当 LRU 缓存空间相对较小时，TCBF_LRU 算法优势更加明显。这是由于当 LRU 缓存较小时，标准 LRU 算法和 BF_LRU 算法，由于缓存空间已满，新到达的大量小流会将 LRU 缓存中的未识别大流信息淘汰掉，而 TCBF_LRU 算法通过时间超时过滤和大流预保护机制，可以将大部分的小流过滤掉，同时保留已识别的大流信息，减少小流对大流的影响，从而减少算法的测量误差。

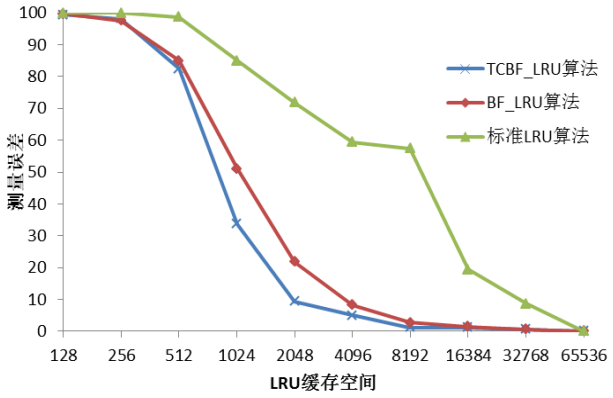

图 10 测量准确性对比图

5 结束语

对于高速网络中大流频繁项监测问题可以用来判定大规模网络中突发性异常和重流量负载链路，以协助 ISP 进行网络基础设备维护和网络资源有效利用，对于短期流量控制和长期流量工程都有重大影响[15]。但传统的网络流频繁项分析算法无法满足在有限的内存空间里，单次线性扫描，在误判率较低的情况下，实现大流频繁项识别。

本文根据传统 LRU 算法容易淘汰大流和频繁更新给系统带来具体压力的缺陷特征，提出一种新的检测大流的 TCBF_LRU 算法，分析了算法的复杂度和误判率，通过模拟数据分析 TCBF_LRU 算法中参数的设置对识别准确性和报文丢弃率的影响。结果表明，TCBF_LRU 算法的检测大流的准确性远高于传统的 LRU 算法和 BF_LRU 算法，而且通过时间超时过滤小流信息，减少 LRU 频繁更新问题，可以实现实时在线处理。

参 考 文 献

[1] 张玉，方滨兴，张永铮.高速网络监控中大流量对象的识别. 中国科学，2010, 40(2): 340-355.

[2] Tatsuya M, Masato U, Ryoichi K. Identifying elephant flows through periodically sampled packets. // Proc of ACM SIGCOMM/IMC'04. Taormina: ACM Press,2004:115-120

[3] 吴桦，龚俭等.一种基于双重 Counting Bloom Filter 的长流识别算法. 软件学报, 2010,21(5):1115-1126

[4] CRISTIAN ESTAN ,GEORGE VARGHESE. New directions in traffic measurement and accounting. ACM TRANSACTIONS ON COMPUTER SYSTEMS, 2003, 21(3): 270-313

[5] Duffield N G, Lund C, Thorup M. Estimating flow distributions from sampled flow statistics. //Proc of the ACM SIGCOMM Conference on Applications, Technologies 2003, Architectures.Karlsruhe: ACM Press, 2003:325-336

[6] 程光，唐永宁.基于近似方法的抽样报文流数估计算法.软件学报，2013,24(2):255-265

[7] KIM S I, REDDY N A L. Identifying long-term high-bandwidth flows at a router. //Proceedings of the 8th International Conference on High Performance Computing. Hyderabad, India, 2001.361-371.

[8] 张震，王斌强等.基于 LRU-BF 策略的网络流量测量算法. 通信学报，2013,34(1):111-120

[9] 王风宇，郭山清等.一种高效率的大流提取方法.计算机研究与发展,2013,50(4):731-740

[10] 杜阿宁，程晓明.网络流量分析中的频繁项监测技术研究.通信学报，2006,27(2):9-15

[11] 王宏，龚正虎. Hits 和 Holds: 识别大象流的两种算法.软件学报，2010, 21(6): 1391-1403.

[12] 周明中,龚俭,丁伟. 网络流超时策略研. 通信学报，2005,26(4):88-93

[13] 王洪波,裴育杰，林宇等.基于 LRU 的大流检测算法. 电子与信息学报. 2007,29(10):2487-2492

[14] 程光，龚俭，丁伟，徐加羚.面向 IP 流测量的哈希算法研究. 软件学报，2005,16(5):652-658

[15] 周爱平,程光,郭晓军.高速网络流量测量方法.软件学报，2014,25(1):135−153.

白磊，男，1982 年生，硕士，讲师，研究方向：网络行为学、网络测量。

田立勤，男，1970 年生，博士，教授，博导，中国计算机学会传感器网络专业委员会委员、Petri 网专业委员会委员，研究方向：网络行为学、网络安全、物联网技术。

陈超，男，1977 年生，博士研究生，副教授，研究方向：并行计算、图形图像处理。