软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2015,26(Suppl.(2)):119−127

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

一种面向用户的企业级无线局域网冗余消除机制*

周新生, 薛广涛

(上海交通大学 计算机科学与工程系,上海 200240)

通讯作者: 薛广涛, E-mail: gt_xue@sjtu.edu.cn

摘 要: 智能手机和移动设备的迅速普及,推动了企业级无线网络的广泛部署.通过收集上海某高校 WiFi 校园网中 10 000 多位匿名用户的 WiFi 网络数据,发现无线局域网中存在超过高达 24%的冗余数据传输.通过分析,挖掘无线用户数据的冗余特性,提出了一种面向用户的冗余数据消除机制.该机制可自适应地根据不同用户冗余度的动态变化,为不同的用户分配不同的缓存大小.从而在路由器缓存有限的条件下获得最大的冗余消除收益.通过基于真实历史数据的仿真实验,证实了该冗余消除机制能够有效识别并消除系统中的冗余数据,并获得较高的系统冗余消除收益.

关键词: 企业级无线局域网;冗余数据消除;缓存分配

中文引用格式: 周新生,薛广涛.一种面向用户的企业级无线局域网冗余消除机制.软件学报,2015,26(Suppl.(2)):119−127. http://www.jos.org.cn/1000-9825/15022.htm

英文引用格式: Zhou XS, Xue GT. User-Oriented redundancy elimination mechanism in enterprise WLANs. Ruan Jian Xue Bao/ Journal of Software, 2015,26(Suppl.(2)):119−127 (in Chinese). http://www.jos.org.cn/1000-9825/15022.htm

User-Oriented Redundancy Elimination Mechanism in Enterprise WLANs

ZHOU Xin-Sheng, XUE Guang-Tao

(Department of Computer Science and Engineering, Shanghai Jiaotong University, Shanghai 200240, China)

Abstract: The rapid spreading of mobile devices and smartphones has promoted the deployment of enterprise WLANs. This research collects read traces from over 10 000 WiFi users in one university in Shanghai. Through intensive data analysis, it finds that there exists more than 24% redundancy traffic in WLANs. By mining the patterns of data set, a user-oriented redundancy elimination strategy is proposed to allocate different cache sizes to different users adaptively. Simulation results show that user-oriented redundancy elimination strategy can detect and eliminate redundant traffic effectively.

Key words: enterprise WLAN; redundancy data elimination; cache allocation

当不同的用户访问相同的网站或同一个用户多次从互联网上获取相同或相似的内容时,相同的数据流将在服务器到用户的数据传输链路上进行多次传输,从而产生了网络中的数据冗余.相关工作[1−4]一直在研究如何高效地消减冗余数据传输.Web 缓存[5]是一种较为成熟的冗余消除解决方案.P2P 缓存机制[6]以其低廉的成本和良好的稳定性也获得了越来越多的关注.但这两种冗余消除技术都是工作在应用层(application-level)上,面向文件和对象的冗余消除称为对象级冗余消除技术.

近些年,越来越多的研究开始关注于协议无关的冗余消除技术,以进一步挖掘冗余消除的潜力.Spring 等人[1]首次提出了协议无关、面向数据流的包级(packet-level)冗余消除算法.该技术被部署在网络层(IP 层),与对象级冗余消除技术相比,它考虑了更小粒度的数据冗余,识别网络数据报中重复出现的连续字节块,因此也被称为包级冗余消除技术.包级冗余消除技术因为不依赖于传输数据包所使用的具体协议,所以是协议无关的,且易

* 基金项目: 国家自然科学基金(61170237)
收稿时间: 2014-05-02; 定稿时间: 2014-08-22

于部署,因而获得了广泛的使用.

包级冗余消除技术的协议无关性使得将其部署在多种网络协议共存的广域网上称为可能,Anand 等人[2−4,7]将包级冗余消除技术部署在广域网的路由器上.Li 等人[8]考虑在 Web 应用数据上添加标识以便冗余消除算法可以更高效的识别并消除冗余.Ditto 等人[9]将协议无关的缓存部署在城域无线网的节点上,获得了更高的数据传输效率.Anand 等人[4]分析了企业级网络中冗余消除的来源,发现带宽的节省主要来自用户自身的流量,用户间的数据冗余仅占总冗余的很小一部分,这证明了传统的非区分用户数据流的冗余消除的共享缓存机制仍有优化的空间.

在本文中,我们主要研究以下几个问题:

1) 收集上海某高校 WiFi 校园网中 2 周的 WiFi 网络数据,数据集中包含 10 000 多位匿名用户通过 WiFi 网络访问互联网的数据信息.总的数据流量唱过 18T.通过对数据集的深入分析,我们发现相对于广域网(WAN)冗余消除情况,无线局域网(WLAN)存在更大的冗余[1,4].同时,无线局域网往往面临更大的网络流量压力,这使得在无线局域网部署冗余消除系统变得迫切.

2) 我们分析了每个用户冗余度和缓存大小的关系,发现面向独立用户的数据传输冗余消除算法可以在较小的缓存上得到较大的冗余消除收益,这也与 Anand 的研究结论[4]相符,即数据冗余主要产生在用户自身的数据流量,而不是用户之间.

3) 我们提出了一种面向用户的端到端的冗余消除机制.该机制可自适应地根据不同用户的冗余度动态变化,为不同的用户分配不同的缓存大小.从而在路由器缓存有限的条件下获得最大的冗余消除收益.通过基于真实历史数据的仿真实验,证实了该冗余消除机制能够有效识别并消除系统中 24%的冗余传输,获得较高的系统冗余消除收益.

1 相关工作

面对日益增长的网络流量与有限的带宽资源,研究人员一直在寻找更高效的冗余消除算法并建立更优秀的冗余消除系统.冗余消除技术通过减少网络中重复传输的数据流达到减少网络流量提高网络传输效率的目的.从冗余识别粒度的角度来看,可以分为对象级冗余消除技术和包级冗余消除技术,两者进行冗余消除的原理和体系结构上存在显著的差异,其中,在包级冗余消除技术中,目前应用最为广泛的是共享缓存体系结构.

1.1 对象级冗余消除技术

Breslau[10]等人的工作证明用户访问 Web 对象的行为符合齐普夫定律,即一个 Web 对象被访问的次数越多,则再次被访问的可能性就越大.因此,将需要频繁访问的 Web 页面和对象保存在离用户更近的系统中,当再次访问这些对象时,不仅能够降低访问时延,而且可以减少网络中的流量.Web 缓存[11,12]是较为成熟的对象级冗余消除技术,是减轻服务器负载,减少客户端访问延迟和降低网络拥塞的有效途径.Web 缓存利用用户访问的时间局部性,比如在短时间内访问相同的网站或从互联网获取相同或相似的内容来达到节省网络带宽的目的,降低访问延迟的目的.它将用户访问过的对象(如图片)缓存起来,当有用户再次请求访问这些对象时,如果缓存内的对象仍是有效的,它会直接响应这些请求,只有当缓存内容失效之后才从内容提供商(content provider)再次获取对象.

1.2 包级冗余消除技术

协议无关的包级别冗余消除技术由 Spring 等人[1]首次提出.这种冗余消除技术能够进一步挖掘冗余消除技术的潜力,在更大程度上减少带宽受限网络链接上的流量.与对象级冗余消除技术不同,包级冗余消除技术关注与网络数据包中连续字节流的冗余,这种冗余消除技术的粒度更小,因此冗余识别率更高,关注数据包的字节流,因此是协议无关的.他们提出了共享缓存结构,对流经路由器的所有用户的数据包运行冗余消除算法,如图 1 所示.将这种冗余消除技术应用到从企业级网络收集的数据集中,Spring 等人[1]发现在下行流量中有 20%的冗余度,在上行流量中有 50%的冗余度.他们还发现,与对象级冗余消除技术相比,包级冗余消除技能能够发现更多

的冗余.

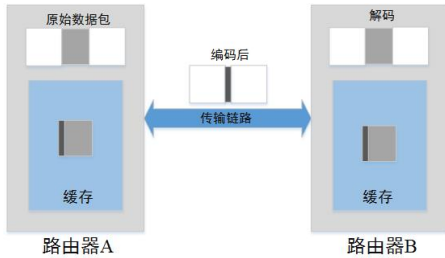


图 1 包级冗余消除技术示意图

1.3 共享缓存体系结构

Spring 等人[1]提出了包级冗余消除技术,将冗余识别的单位由对象级冗余消除技术的 WEB 图片,WEB 视频,或文件减小到大小仅有数十字节的网络数据包的连续字节流.他们将冗余消除系统部署在 IP 层,对 IP 数据包中 IP 和 TCP 协议首部以外的负载(payload)数据部分进行冗余识别和消除.在图 1 所示的结构中,通信链路的两段是对称的,它们可以是相邻的路由器也可以是网络连接的两段.链路的一端发送数据包之前先去缓存中查看数据包的负载部分是否有冗余,假如有冗余,则将冗余部分编码成较短的冗余标识传到链路的另一个端.同样,链路的另一端对收到的数据包进行解析,将编码后的冗余部分还原成对应的数据部分.为了保证收发双方能够正确地完成数据包的编码和解析,需要同步收发双方的缓存.此外,Spring 等人[1]的缓存体系将流经链路两段的所有数据包不作区分,视为一股数据流,这股数据流可以来自多个用户,因此属于共享缓存.

2 数据集

我们收集了上海某高校无线网络从 2013 年 1 月 1 日~2013 年 1 月 14 日 10 000 多位师生的匿名数据,数据集来自部署在 22 栋建筑物的 334 个访问节点(access point)中超过 10 000 多用户的所有数据包.数据集还包括用户每次连接到无线网的时间戳、AP 标识、离开无线网的时间戳以及被分配到的动态 IP,使我们能够将流经路由器的所有数据包按照用户分类,见表 1.

表 1 数据集信息

第几天	时长(h)	用户数	数据总量(GB)
1	9	7 797	475
2	24	7 961	1 640
3	24	7 533	1 577
4	24	5 479	854
5	24	4 524	1 274
6	24	4 701	1 269
7	24	3 465	124
8	24	5 841	1 732
9	24	5 871	2 000
10	24	5 563	2 171
11	24	4 648	1 735
12	24	4 940	1 974
13	24	6 013	1 710
14	11	5 912	352

上海某高校无线校园网属于典型的企业级无线局域网,包括 2 000 多个访问点(AP)、一个无线局域网控制器和一个后端的数据中心.无线网络控制器负责管理 AP,将新加入的用户分配到不同的 AP 上,同时兼顾负载均衡.我们从后端的数据中心(data center)收集数据包和对应的登录信息.

3 冗余计算

冗余识别是包级冗余消除技术的第 1 步,只有将冗余从用户数据中识别出来才能进行后面的编码工作.我

们将流经路由器的所有数据包按照用户分类,然后对每一个用户的所有数据包使用 Rabin Karp[13]算法计算数据包的指纹,由于路由器缓存空间的限制,无法将所有的指纹放到缓存中[14],因此需要对采集到的指纹抽样,仅选取那些代表性较好的指纹放入路由器缓存.

3.1 指纹计算

对于流经路由器的所有数据包,我们需要从数据包的负载中快速找出曾经出现过的部分.对于数据负载部分连续的β个字节 $t_1,t_2,t_3,\ldots,t_\beta$,Rabin 指纹的计算公式如下:

$$RF(t_1,t_2,t_3,t_4,t_5,\ldots,t_\beta)=(t_1p^\beta+t_2p^{\beta-1}+\ldots+t_{\beta-1}p+t_\beta) \bmod M,$$

其中,p 和 M 都是定值,窗口大小β作为参数可以调节.假设我们已经计算出 $RF(t_1,t_2,t_3,\ldots,t_\beta)$,那么,$RF(t_2,t_3,\ldots,t_{\beta+1})=(RF(t_1,t_2,t_3,\ldots,t_\beta)-t_1p^\beta)*p+t_{\beta+1}$.

从 Rabin 指纹[13]的计算公式中可以看出,算法支持增量计算,使得计算数据包指纹序列的时间复杂度接近线性,为了进一步加快指纹的计算速度,我们可以先将 p^β的值保存下来,计算过程如图 2 所示.

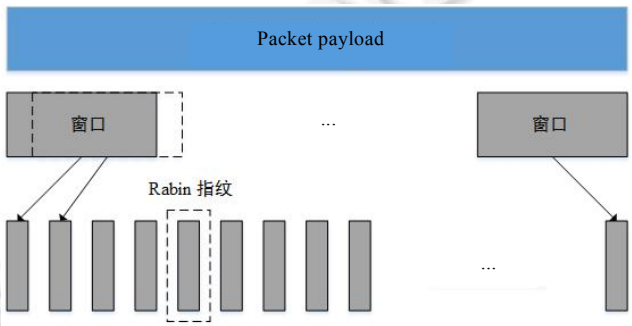


图 2 计算 Rabin 指纹示意图

使用 MD5[15]和 SHA[16]同样能够为数据包生成指纹,但与 Rabin Karp 算法相比,这两种算法计算复杂度较高且不支持增量计算,考虑到路由器有限的计算能力,Rabin Karp 算法是较好的选择.

3.2 指纹采样

Rabin Karp 算法要求指纹计算的连续性,即每隔一个字节就要计算一个指针.假如一个用户某一天产生了 1G 的数据流量,对这些数据每隔一个字节计算一个指纹,那么产生的指纹将数十亿计,即使每个指纹的存储空间只有 4 字节,仅存储指纹就需要 4G 的空间.

在实际处理过程中,我们仍然按照 Rabin Karp 的方法计算指纹,但选取指纹放入缓存的策略是每隔 n 个指纹放入一次.考虑这一种情况,有两个数据包,第 1 个数据包的负载(payload)比第 2 个数据包多出 1 个字节,而两个数据包的其他部分完全相同,那么上述的采样算法计算出的冗余度是 0,显然这种方法是不合理的[17].

3.2.1 MODP 采样算法

Manber[18]等人的解决办法是对于 Rabin Karp 算法生成的指纹序列,按照指纹的值,而不是指纹的位置来决定哪些放入缓存.Rabin Karp 能够保证选取的值分布均匀,代表性好.MODP 算法就是选取那些对一个定值 p(P 作为一个参数可以由研究人员给定)取模得 0 的指纹放入缓存.用这种方法选取的指纹独立于指纹的位置,不会因为数据包是倒序或者有额外的一个字节造成冗余度计算的偏差.参数 p 决定了存储指纹缓存的大小,一般来讲,指纹 p 大小在 32~128 之间,比如在文献[15]中,每隔 1 500 字节取 16 个指纹,相当于 p 等于 90.

3.2.2 Winnowing 采样算法

MODP 算法的一个缺点是指纹选取具有全局性的特征,对全局参数的依赖可能导致样本不足或过度采样的问题.Winnowing[19]算法对指纹序列采集局部最小值作为样本.Winnowing 在每个窗口内都选择最小的哈希值作为样本.当一个窗口内有多个最小值时,选择最右边的.与 MODP 采样算法相比,Winnowing 算法能够实现样

本的均匀分布.

在实际的处理过程中,我们分别应用 MODP 和 Winnowing 算法对指纹进行采样,发现使用 Winnowing 采样算法能够获得比 MODP 更高的命中率.更高的命中率意味着更多的流量节省,在本文中,我们采样 Winnowing 算法作为我们的指纹采样算法.

4 模型与方法描述

4.1 面向用户的冗余消除机制系统架构

我们发现,为每个用户独立进行冗余数据消除算法可以在较小的缓存开销下获得较大的冗余消除收益,即数据冗余主要产生在用户的自身数据传输过程中,用户间的数据冗余并不明显[4].受此启发,我们提出了独立缓存冗余消除技术,基于动态规划算法,在存储空间有限的路由器上为冗余消除收益更高的用户优先分配缓存,以使流量节省最大化.系统架构如图 3 所示.

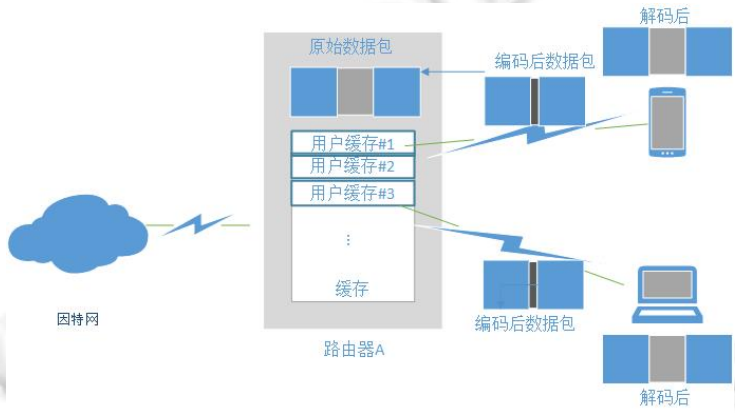


图 3 面向用户的冗余消除系统架构

4.2 计算用户合适的缓存大小

我们定义函数 CL 来计算某个缓存尺寸下对应的冗余度,那么对于某个用户 i,

$$traffic_{\max}=CL(InfiniteMemory),$$

其中,*InfiniteMemory* 表示分配给用户 i 足够大缓存,在此缓存空间下,可以得到用户 i 的最大流量节省比例 $traffic_{\max}$.我们定义用户 i 的合适缓存大小为 $fitsize_i$:

$$\text{Minimize } fitsize_i,$$

$$\text{Subject to CL}(fitsize_i)\geqslant traffic_{\max}-3\%.$$

这样,对于用户 i 的合适缓存空间,兼顾了冗余消除效率和尽可能小的缓存大小.

算法 1. Calculate fit cache size for user i.

输入: All packets from user i.

输出: fit cache size for user i, $fitsize_i$.

1. Initialize a big cache size $size_{\max}$ which is enough to hold all fingerprint of user i
2. Calculating corresponding traffic saving $saving_{optimal}$ when cache size is $size_{\max}$
3. Set $size_{\min}$ and $saving_{\min}$ to 0
4. **while** $saving_{\min}<saving_{optimal}-0.03$
5. $fitsize_i=(saving_{\min}+saving_{optimal})/2$
6. calculate $saving_{mid}$ when cache size is $fitsize_i$
7. **if** $saving_{mid}<saving_{optimal}-0.03$

8. $size_{\min}=fitsize_i+1$
9. **else**
10. $size_{\max}=fitsize_i-1$
11. **end if**
12. **end while**
13. $fitsize_i=size_{\min}$

我们先为第 i 个用户分配一个较大的缓存空间,并计算其最优的流量节省比例.然后使用二分法不断调节缓存大小直到某个缓存大小 $fitsize_i$,此时,用户的流量节省与最优流量节省相差小于 3%,且缓存大小最小.

我们使用 Rabin Karp 算法和先进先出缓存更新策略来计算给定一个缓存大小对应的流量节省,使用二分法调节缓存尺寸,这样能够高效地算出期望值.

4.3 独立缓存体系结构算法实现

文献[1,20]的共享缓存体系结构将流经路由器的所有数据包视为一股数据流,对这股数据流运行冗余消除算法.文献[4]对冗余来源做了比较详细的分析,并发现大部分流量节省来自用户各自流量内部的冗余,而用户流量之间的冗余仅占少部分.受文献[4]工作的启发,我们提出一种独立缓存体系结构,将流经路由器的数据包按照用户分开(说明),为每一个用户分配缓存.

在为每一个用户计算出合适的缓存大小后,我们要为用户分配缓存.在理想情况下,我们可以为每一个用户分配缓存,但考虑到实际情况中,路由器的存储空间有限,我们需要一个遴选的过程.我们将问题描述转化为数学模型,路由器的存储空间为 $size_{\max}$,对于流经路由器的用户 i,我们已经计算出对应的 $fitsize_i$,$saving_i$ 参数,注意,这里我们将 $saving_i$ 设置为用户 i 对应的流量节省的字节数不再是节省比例.我们的目的是在 N 个用户中遴选出 K 个,使得这 K 个用户所需要的缓存空间小于 $size_{\max}$,并且流量节省最大,满足:

$$\text{Maximize } \sum_{i=1}^{n} saving_i,$$

$$\text{Subject to } \sum_{i=1}^{n} fitsize_i \leqslant size_{\max}.$$

我们使用动态规划来解决这个问题.我们假设 $f[i][v]$表示为前 i 个用户分配大小为 k 的缓存空间能够得到的最大流量节省,则动态规划解法的状态转移方程是

$$f[i][v]=\max[f[i-1][v],f[i-1][v-c[i]]+w[i]],$$

其中,v 表示缓存空间的大小,$w[i]$表示第 i 个用户的合适缓存大小(fit cache size).

对于为前 i 个用户分配缓存这个子问题,若只考虑是否为第 i 个用户分配缓存(分或不分),就可以转化成一个只牵扯为前 $i-1$ 个用户分配缓存的问题.如果不为第 i 个用户分配缓存,那么问题就转化为"为前 $i-1$ 个用户分配大小为 v 的缓存空间";为第 i 个用户分配缓存,那么问题就转化为"为前 $i-1$ 个用户分配剩下的容量为 $v-c[i]$ 的缓存空间",此时能够获得的最大流量节省就是 $f[i-1][v-c[i]]$再加上通过为第 i 个用户分配缓存空间带来的流量节省 $w[i]$.

算法 2. Choose user group who save more traffic data.

输入: Router cache size M, $\langle fitsize_i,savingbytes_i\rangle$ pairs for each user, user number N.

输出: User list $List$[] that will be allocated cache at router.

1. **for** $i\leftarrow 1$ to N
2. **for** j$\leftarrow 1$ to M
3. **if** $c[i-1][j]>c[i-1][j-fitsize_i]+savingbytes$
4. $u[i][j]=u[i-1][j]$
5. **else**
6. $u[i][j]=u[i][j-fitsize_i]+user_i$
7. **end if**

8. **end for**

9. **end for**

10. *List*[]=*u*[*N*][*M*]

值得注意的是,假如为 *K* 个用户分配缓存后,路由器的缓存还有部分剩余,那么我们就从剩下的还未分配缓存的用户组中选取流量节省比最大的,补充到路由器缓存中.

5 性能评估

5.1 参数选取

参数λ,β分别代表计算指纹时,指纹的长度和采样率.λ,β两个参数的选取对冗余度的计算影响很大,假如λ太大,则意味着较长的连续字节块作为冗余识别的基本单位,在这种情况下,一旦成功地识别出一块冗余,就会得到较高的流量节省,但有可能无法识别一些长度较小的冗余块.采样率β影响用户缓存的大小,β越大,采样率越高,会有更多的指纹放入缓存,会识别更多的冗余,但路由器的缓存大小有限,采样率β的大小要保证在一定范围内.λ,β参数的选择是计算量,存储空间和流量节省之间的权衡.根据文献[1]的研究推荐,我们在实际的运算过程中,将λ设置为 64 字节,β设置为 128 字节,即每隔 128 字节选取一个 64 字节长度的连续字节块放入缓存.并在实验中取得较好的效果.

5.2 数据集特征

对于数据集的特征,我们首先要考虑的是数据集本身的冗余度如何,计算冗余度时,我们为数据分配较大的缓存空间,即,假设缓存的大小是没有限制的.用户冗余数据的 CDF 图如图 4 所示.

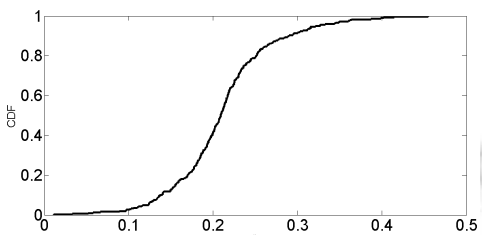


图 4 用户冗余度 CDF 图

我们发现在企业级校园无线网络中,下行流量占到了总数据量的 80%,且下行冗余度较高,达到了 30%.在对知名端口冗余度的统计中,HTTP 产生的流量最多且冗余度最大.从表 2 可以看出,冗余度与上下行流量、协议类型都有较为明显的关系,这对我们设计更优秀的缓存更新策略具有指导意义.

表 2 数据集冗余度

服务类型	流量比例(%)	冗余度(%)
上行	23.4	19.12
下行	76.6	30.32
TCP	80.35	30.76
UDP	19.5	18.72
HTTP	71.4	24.46
HTTPS	3	0.62
DNS	0.5	0.12
总体	–	24.75

5.3 用户合适缓存大小

我们从数据集中随机抽取 300 名用户所产生的数据包,并为每个用户计算其合适的冗余缓存大小.冗余缓存的更新策略为先进先出(FIFO),缓存中存储采样后的指纹以及指纹对应的真实数据(64 字节的连续字节流).

图 5 展示了对于 300 个用户对应合适缓存大小的分布.从图中可以看出,80%用户的合适冗余缓存大小都在 10MB 以内(图 5 的横坐标是对缓存大小取对数之后的值).因此,优先为那些流量节省缓存大小比较高的用户分配缓存将会获得较好的冗余消除收益.

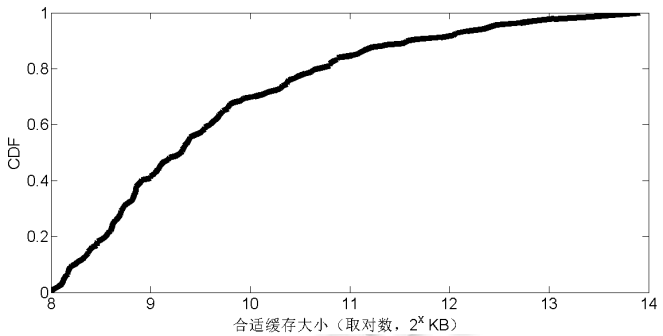


图 5 用户合适缓存大小 CDF 图

5.4 独立缓存体系结构的流量节省

我们对比两种冗余消除技术.第 1 种为随机选取部分用户为其分配缓存空间,第 2 种为基于动态规划解法选取的用户分配缓存空间,如图 6 所示.

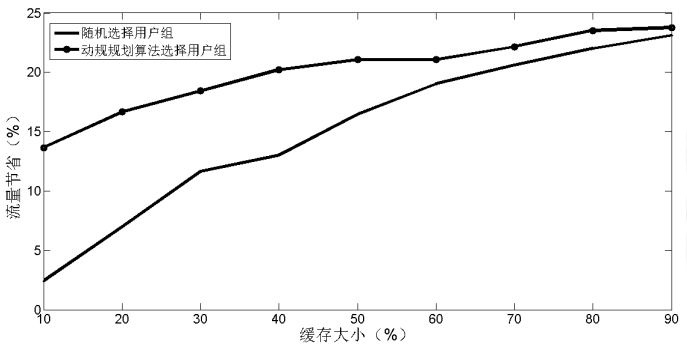


图 6 缓存分配策略与冗余度的关系

首先,我们为流经路由器的所有用户计算其对应的合适缓存大小(fit cache size),得到每个用户的合适缓存大小为 $cachesize_{optimal}$,这意味着,假如路由器的缓存大小为 $cachesize_{optimal}$ 或更大,我们可以为所有用户分配缓存.考虑实际中路由器缓存大小有限,不可能完全满足所有的用户冗余缓存分配,我们将缓存大小分别设置为 $cachesize_{optimal}$ 的 10%,20%,...,90%,然后分别运行两种缓存分配算法,结果如图 6 所示.

从图 6 我们可以看出,当缓存大小极为有限时(10%的 $cachesize_{optimal}$),基于动态规划选择的用户组比随机选取用户组的优势非常明显,这是因为动态规划的策略是优先选取流量节省最大的用户组合.随着缓存大小的增加,两种选择策略之间的差距越来越小,当缓存大小增加到 $cachesize_{optimal}$ 时,两者的流量节省相同.

6 总 结

Sping 等人[1]提出包级冗余消除机制冗余识别粒度更小且协议无关,与传统的对象级冗余消除技术相比,包级冗余消除技术面向数据包中连续的字节流,冗余识别的粒度更小,因此能够消除更多的冗余;包级冗余消除技术关注的是字节流,因此是独立于协议的.这两种特质使包级冗余消除系统得到了广泛的应用.

我们收集了上海某高校 1 万余名师生两周的匿名数据.通过对数据集的深入研究,我们发现了两个有趣的数据特征:首先,相对于广域网而言,企业级无线局域网存在更大的数据传输冗余,同时企业级无线网络往往面

临更大的网络流量压力,这使得在企业级无线网络应用冗余消除技术变得更为迫切;其次,通过验证发现,为每个用户独立进行冗余数据消除算法可以在较小的缓存开销下获得较大的冗余消除收益,即数据冗余主要产生在用户的自身数据传输过程中,用户间的数据冗余并不明显.受此启发,我们提出了独立缓存冗余消除技术,基于动态规划算法,在存储空间有限的路由器上为冗余消除收益更高的用户优先分配缓存,以使流量节省最大化.通过基于真实历史数据的仿真实验,本文证实了所提出的冗余数据消除机制能够有效识别并消除系统中 24%的冗余传输,获得较高的系统冗余消除收益.

References:

[1] Spring N, Wetherall D. A protocal-independent technique for eliminating redundant network traffic. ACM SIGCOMM Computer Communication Review, 2000.

[2] Anand A, Gupta A, *et al*. Packet caches on routers: The implications of universal redundant traffic elimination. ACM SIGCOMM Computer Communication Review, 2008,38(4):219−230.

[3] Anand A, Sekar V, Akella A. SmartRE: An architecture for coordinated network-wide redundancy elimination. In: Proc. of the SIGCOMM. ACM, 2009. 87−98.

[4] Anand A, Muthukrishhan C, Akella A, *et al*. Redundancy in network traffic: Findings and implications. In: Proc. of the 11th Int'l Joint Conf. on Measurement and Modeling of Computer Systems. 2009.

[5] Squid Web proxy cache. http://www.squid-cache.org/

[6] PeerApp: P2P and media caching. http://www.peerapp.com

[7] Agarwal B, Akella A, Anand A, *et al*. EndRE: An end-system redundancy elimination service for enterprises. In: Proc. of the NSDI. 2010.

[8] Li X, Salyers D, *et al*. Improving packet caching scalability through the concept of an explicit end of data marker. In: Proc. of the HotWeb. 2006.

[9] Dogar F, Pucha H, *et al*. A system for opportunistic caching in multi-hop wireless mesh network. In: Proc. of the ACM MobiCom. 2008.

[10] Breslau L, Cao P, *et al*. Web caching and Zipf-like distributions: Evidence and implications. In: Proc. of the IEEE INFOCOM. 1999.

[11] Wolman A, *et al*. On the scale and performance of cooperative Web proxy caching. In: Proc. of the ACM Symp. on Operating Systems Principles. 1999.

[12] Qian F, Quah KS, Huang J, *et al*. Web caching on smartphones: Ideal vs. reality. In: Proc. of the 10th Int'l Conf. on Mobile Systems, Applications, and Services. 2012.

[13] Rabin MO. Fingerprinting by random polynomials. Technical Report, TR-15-81, Center for Research in Computing Technology, Harvard University, 1981.

[14] Ziv J, Lempel A. A university algorithm for sequential data compression information theory. IEEE Trans. on Information Theory, 1997,23(3):337−343.

[15] Rivest R. The MD5 Message-Digest Algorithm. RFC 1321, Networking Working Group, Requests for Comment, 1992.

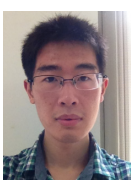
[16] Specifications for Secure Hash Standard. National Institute of Standards and Technology, 1995.

[17] SAHA S. On reducing the processing load of redundancy elimination algorithms. In: Proc. of the GLOBECOM Workshops (GC Wksps). 2011.

[18] Manber U. Finding similar files in a large file system. In: Proc. of the USENIX Winter Technical Conf. 1994.

[19] Schleimer S, Wilkerson D, Aiken A. Winnowing: Local algorithm for document fingerprinting. In: Proc. of the 2003 ACM SIGMOD Int'l Conf. on Management of Data. 2003.

[20] Fan L, Cao P, Almeida J, Broder AZ. Summary cache: A scalable wide-area Web cache sharing protocol. IEEE/ACM Trans. on Networking, 1998. 254−265.



周新生(1991－),男,山东济宁人,硕士,主要研究领域为无线网络,移动计算.

薛广涛(1976－),男,博士,副教授,博士生导师,CCF 高级会员,主要研究领域为无线网络,移动计算,分布式计算,大数据.