·体系结构与软件技术· 文章编号：1000-3428(2019)05-0046-06 文献标志码：A 中图分类号：TP393

基于节点介数与边缘流行度的 NDN 缓存策略

陈劼博，郑 烇，王 嵩

（中国科学技术大学 自动化系 未来网络实验室，合肥 230026）

摘 要： 针对命名数据网络（NDN）架构中多数缓存策略的冗余与低效问题，提出一种基于节点介数与边缘内容流行度的缓存放置策略 BEP。结合节点的介数中心性与内容的动态流行度，同时考虑缓存的过滤作用，将最流行的内容放置在最重要的节点上，以高效利用稀缺的缓存资源。仿真结果表明，与经典 NDN 缓存策略 LCE、LCD 相比，BEP 能有效提高缓存命中率，降低服务器负载。

关键词： 命名数据网络；缓存；节点中心性；流行度；缓存决策；网络边缘

中文引用格式： 陈劼博，郑烇，王嵩. 基于节点介数与边缘流行度的 NDN 缓存策略[J]. 计算机工程，2019，45(5)：46-51.

英文引用格式： CHEN Jiebo，ZHENG Quan，WANG Song. NDN caching strategy based on node median and edge popularity[J]. Computer Engineering，2019，45(5)：46-51.

NDN Caching Strategy Based on Node Median and Edge Popularity

CHEN Jiebo，ZHENG Quan，WANG Song

（Laboratory for Future Networks，Department of Automation，University of Science and Technology of China，Hefei 230026，China）

【Abstract】 Aiming at the redundancy and inefficiency of most caching strategy in Named Data Networking（NDN）architecture，a caching placement strategy BEP based on node median and edge content popularity is proposed. Combining the median centrality of node with the dynamic popularity of content，and considering the filtering effect of caches，the most popular content is placed on the most important node to make efficient use of scarce cache resources. Simulation results show that compared with the classical NDN caching strategies LCE and LCD，BEP can effectively improve the cache hit rate and reduce the server load.

【Key words】 Named Data Networking（NDN）；caching；node centrality；popularity；caching decision；network edge

DOI： 10.19678/j.issn.1000-3428.0050182

0 概述

Cisco 公司的 VNI 预测报告指出[1]，未来五年内全球 IP 流量将会在 2017 年的基础上翻三番，且视频流量将占据越来越大的比例。命名数据网络（Named Data Networking，NDN）[2]是 ICN（Information Centric Networking）[3]的主要实现方式之一，泛在的网络缓存使其能够快速响应用户请求，降低服务器负载，减少响应时间，并在一定程度上减少带宽的使用，最终以较低的存储空间获取时间与带宽资源。NDN 默认的缓存策略 LCE（Leave Copy Everywhere）[4]将内容缓存在数据包经过的所有节点上，该策略会产生大量的内容冗余副本，且多数副本不会被再次请求，最终造成了极大的资源浪费。文献[5]提出的 Betw 策略将内容放置在介数最大的节点上，但其没有考虑高替换率问题[6]，从而造成热点内容被过早替换的现象。

为解决 NDN 架构中缓存策略的冗余与低效问题，本文综合考虑内容的动态流行度、节点在拓扑结构中的重要程度，以及缓存对用户请求的过滤作用，提出一种基于节点介数与边缘内容流行度的缓存放置策略 BEP（Betweenness and Edge Popularity）。将最流行的内容放置在最重要的节点上，利用稀缺的缓存资源来避免大量的冗余，同时解决 Betw 策略的高替换率问题，从而提升整个缓存系统的性能。

1 相关技术

NDN 缓存主要分为缓存放置策略和缓存替换策略。前者决定将内容放置在哪些节点，后者决定替换节点中的哪些内容。本文主要研究缓存放置策略。在缓存放置策略中，研究者们更加倾向于利用节点或网络信息进行协作缓存，协作方式分为显示协作和隐式协作。虽然显示协作[7-9]的缓存放置策略能够降低缓存系统的内容冗余度，确保缓存多样

基金项目： 国家自然科学基金重点项目“三网融合业务接入系统的分析、建模与调控”（61233003）。

作者简介： 陈劼博（1993—），男，硕士研究生，主研方向为未来网络架构、网络缓存技术；郑 烇（通信作者）、王 嵩，副教授。

收稿日期： 2018-01-19 **修回日期：** 2018-03-16 **E-mail：** qzheng@ustc.edu.cn

性,提高缓存收益,但是需要额外的信息交互和计算工作,即需要大量的计算和网络资源。不同于显示协作,隐式协作无需提前获取网内节点的状态信息,其主要策略有介数 Betw[5]、MCD[10]、LCD[11] 以及 ProbCache[10] 等。

NDN 默认的缓存放置策略是 LCE[4],该策略算法简单、运算速度快,但会造成大量的内容冗余[7],且大部分内容在被用户再次请求之前就被替换掉,成为一次无效缓存。

LCD 策略[11]每次将内容缓存在命中节点的下一跳,即每命中一次,内容向用户移动一跳。该策略能够隐式利用内容的请求频率,将请求频率较高的内容移到距离用户较近的节点上,从而解决 LCE 的高冗余度问题。但由于其移动速度较慢,内容很有可能在中途节点被替换掉,使用户不得不重新从上游节点获得内容。MCD 策略[10]与 LCD 思想基本一致,只是在内容向下游移动之后,MCD 策略会删除上游节点中的副本。MCD 的冗余度比 LCD 低[10],但其内容一旦被替换掉,用户就不得不重新从源服务器重新获取内容。

概率缓存[12]是一种简单高效的缓存策略,其在数据包经过每个节点时以概率 p 缓存该数据包,在一定程度上解决了 LCE 的高冗余度问题,但该策略具有一定的随机性,使其广泛应用受到限制。

文献[13]提出的 ProbCache 策略综合考虑数据包距离用户的距离和节点可用缓存的大小,数据包越接近用户节点,内容被缓存下来的概率就越大,同时缓存容量充足的节点有更高的概率缓存内容。但 ProbCache 策略可能会引起边缘节点的缓存竞争,造成节点替换率过高、命中率过低的结果。

文献[5]提出的 Betw 策略利用了复杂网络中的介数概念[14]。用节点的介数来衡量节点在网络拓扑中的重要程度,其数学定义如下:

$$C_B(v)=\sum_{s\neq v\neq t\in V}\frac{\sigma_{s,t}(v)}{\sigma_{s,t}} \tag{1}$$

其中,V 是网络拓扑中所有节点的集合,$C_B(v)$ 是节点 v 的介数,s、t 是不同于 v 的另外 2 个节点,$\sigma_{s,t}$ 是节点 s 到 t 所有最短路径的数量,$\sigma_{s,t}(v)$ 是 s 到 t 的最短路径中经过节点 v 的数量。从式(1)可以看出,节点的介数越大,经过该节点的路径越多,该节点在网络中成为一个枢纽节点的可能性越大。Betw 策略将内容缓存在兴趣包经过的介数最大的节点上,在降低冗余度的同时,利用该节点高介数的特性,使得更多的其他节点能够快速地获取内容。但该策略的主要缺点是没有考虑高替换率问题[6],若所有内容都放置在介数最高的节点上,缓存容量的限制将会造成缓存替换,最终热点内容可能被替换出缓存,而非热点内容将占据缓存。

分级缓存策略[15]将缓存和内容根据请求和跳数进行分级,并按照级别放置,该策略未考虑边缘节点对请求的过滤作用,导致分级不准确。文献[16]同时考虑节点热度和缓存替换率,但其仅将请求数量作为节点的热度评判标准,使得缓存集中在源节点周围。文献[17]从缓存收益的角度出发,以随机的方式将高收益内容以高概率进行缓存,该策略是对概率缓存[12]和 ProbCache[13]的一种改进,但其仍无法解决边缘节点的缓存竞争问题。

现有关于 NDN 网络缓存的研究大都仅考虑内容特征或拓扑特征,且未对缓存的过滤作用进行探索。将内容特征和拓扑特征相结合,能够利用更多的信息并更合理地放置网络内容,从而提高缓存命中率,降低服务器负载,最终提升用户体验。

2　缓存放置策略 BEP

为提高缓存空间的利用率,降低服务器负载,本文结合内容的动态性、流行度以及节点在拓扑中的重要程度,提出一种缓存放置策略 BEP。在网络的边缘周期性地统计每个内容的流行度,根据流行度的排名和沿途节点的重要程度决定内容的缓存位置。BEP 的核心思想是将最流行的内容缓存在最核心的节点上,非流行内容放置在次要节点上,从而解决 Betw 中所有内容都放置在核心节点上所造成的高替换率问题,使得核心节点缓存流行的内容,最终对稀缺的缓存资源进行更高效的利用。

2.1　边缘动态流行度统计

在树状拓扑中对于 LRU(Least Recently Used)策略,一个缓存节点相当于一个低通滤波器[18],节点缓存高频内容并服务相应的请求,低频内容将会通过该节点向上转发。因此,在树状拓扑中,对上层节点的请求统计并不能反映所有内容的真实请求频率,以此频率计算统计内容流行度将不具备代表性与全局正确性。

运营商在边缘网络中的典型拓扑就是树状拓扑,因此,本文针对树状拓扑提出一种在边缘节点(即叶子节点)统计并计算内容流行度的算法。内容流行度即用户对内容的请求次数,本文算法周期性地统计每个内容在每个叶子节点上的请求次数,并计算相应的流行度。内容 c 在节点 n 的第 i 个计时周期内流行度的计算公式为:

$$Popularity_{c,n}(i)=\alpha\times Popularity_{c,n}(i-1)+(1-\alpha)\times N_{c,n}(i) \tag{2}$$

其中,α 是衰减因子,$0<\alpha<1$,其表示本周期之前该内容流行度所占的比例,$N_{c,n}(i)$ 表示内容 c 在节点 n 的第 i 个计时周期内被请求的次数。距离当前周期越远的周期,请求次数在流行度中所占的比例就越小。式(2)是较常用的流行度计算方法,能够动态适应内容流行度的变化,较好地反映近期的热点内容,从而对缓存对象进行准确判断。

为实现对动态流行度的统计和计算,本文在NDN网络节点中加入一个流行度表,如表1所示。假设 α 取值为0.75,已知内容的历史流行度和当前周期被请求的次数,可以通过式(2)计算出当前的内容流行度。

表1 边缘节点流行度表

内容	历史流行度	本周期内请求次数	当前流行度
/video/1.mpg	25.24	30	26.43
/video/2.mpg	16.32	20	17.24
/video/3.mpg	14.23	10	13.17
/video/4.mpg	9.16	0	6.87
⋮	⋮	⋮	⋮

2.2 缓存决策算法

在网络拓扑中,不同节点的重要程度不同,某些节点能够联通较多其他节点,可以认为这种节点在网络拓扑中属于重要节点,有较多的网络请求会途经这类节点。如果将最流行的内容放置在最重要的节点上,将有很多的请求会在重要节点得到响应而不会产生更多的转发,最终提高全网命中率并降低服务器负载。

在CCN网络中,与其他中心性相比,介数中心性具有较低的服务器部署成本与较短的请求时延[19]。因此,本文选择介数中心性作为节点重要性的度量。在边缘节点进行内容流行度统计之后,需要根据内容排名和节点缓存能力大小将内容放置在合适的节点上。

用户产生兴趣包后,根据相应的转发策略将兴趣包转发给服务器,在第一跳即边缘节点,BEP策略会根据内容流行度表获取该内容的流行度排名并添加到兴趣包中。另外,BEP策略会记录沿路所有节点的介数大小,在缓存命中或者兴趣包到达服务器时,当前节点根据兴趣包中的信息找出合适的节点,并将该节点的介数添加到返回的数据包中,数据包在返回过程中与沿途节点的介数逐一比较,当节点介数与数据包中介数相等时,就将此数据缓存在当前节点。设节点缓存能力大小为 $CacheSize$,内容 c 在边缘节点的流行度排名为 $Rank_c$,则应放置节点的介数在沿途所有节点介数中的排名 $R_{\text{betweenness}}$ 为:

$$R_{\text{betweenness}}=\left\lfloor\frac{Rank_c}{CacheSize}\right\rfloor \tag{3}$$

BEP算法伪代码如下:

算法1 BEP算法

```
初始化(rank = 0,bool isEdge = false,betw = 0)
节点 v 收到请求 interest:
if data in cache
rank = interest.getRank();
r = rank/cache_size;
betws[] = interest.getBetws();//获取介数数组
betw = getRthBetw(r,betws);//得到第 r 大介数
data.addField(betw);
send(data);
else if node v is server and provide data
rank = interest.getRank();
r = rank/cache_size;
betws[] = interest.getBetws();//获取介数数组
betw = getRthBetw(r,betws);//得到第 r 大介数
data.addField(betw);
send(data);
else
isEdge = isEdgeNode(v); //判断节点 v 是否为边缘节点
if isEdge
rank = v.getRank(interest.getName());
interest.addField(rank);
end if
betw = getBetw(v);//获取当前节点介数
interest.appendBetw(betw);//将介数添加进兴趣包
forward interest to the next hop;
end if
节点 v 收到请求 data:
betw = data.getBetw();
current_betw = getBetw(v);
if(betw = = current_betw)
cache(data);
end if
forward data to the next hop;
```

介数仅与拓扑结构相关,因此,本文提前计算好所用介数并存入各节点中。如图1所示,在网络拓扑中共有3个用户和一个服务器节点,假设中间5个网络节点的缓存大小为1。由式(1)可算得各节点的介数大小,图1中节点下方数字代表各自的介数值,假设节点 v_3 的流行度表为表1。在初始时刻,v_1 ~ v_5 5个节点缓存都为空。若此时用户 A 请求名为/video/1.mpg的内容,边缘节点 v_3 收到此兴趣包后,通过查询流行度表可知,该内容流行度排名为1,因此将兴趣包的 $Rank_c$ 设置为1,并将 v_3 的介数值0.25加入兴趣包的介数数组中,然后将兴趣包转发至 v_2 节点。v_2 节点收到该兴趣包后,将自己的介数值0.71加入介数数组中。最终兴趣包到达服务器 S。服务器 S 获得兴趣包的 $Rank_c$ 值和介数数组[0.61,0.71,0.25],由 $CacheSize=1$ 和式(3)可以算得 $R_{\text{betweenness}}=1$,因此,从介数数组中找到最大值0.71放入返回的数据包中。数据包返回时逐一比较沿途节点的介数,当介数等于0.71时就将数据缓存在该节点。在图1的示例拓扑中,内容将被缓存在节点 v_2 中。若用户 A 请求内容/video/2.mpg,则 $R_{\text{betweenness}}=2$,此时返回数据包将携带排名第2的介数0.61,节点 v_1 会缓存这一内容。

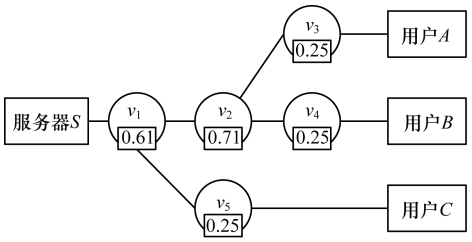

图 1　示例拓扑

3　实验结果与分析

3.1　实验环境与参数设置

本文选用 ndnSIM[20-22] 作为网络仿真平台以验证 BEP 策略的性能。实验环境设置:操作系统为 MacOS High Sierra 10.13,内存为 4 GB,CPU 为 Intel Core i5 16 GHz,ndnSIM 版本为 2.4。仿真拓扑采用树状结构,共 7 层,从根节点开始随机生成子节点,每个节点有 0 ~ 5 个子节点,最终生成 628 个节点。其中,设置 1 个源服务器节点(树的根节点),377 个用户节点(所有叶子节点)。网络内共有$N=2\ 000$ 种不同的内容,每个内容大小都相同,为1 024 KB,内容流行度服从 Zipf-Mandelbrot 分布[23],Zipf 参数范围为 0.5 ~ 1.5。每个用户节点的请求都服从泊松分布,且频率 $\lambda=10$ req/s。每个节点的缓存大小都相同且为 C,C 的取值范围为10 个 ~ 100 个内容,使得内容存储(Content Store,CS)表缓存大小与内容总量之比 $R(R=\frac{C}{N})$控制在 0.005 ~ 0.050 之间,以确保节点缓存容量远小于内容总量,符合网络真实情况。在 BEP 算法中,统计边缘流行度时衰减因子 α 取值为 0.75,统计计算周期为 1 s,仿真时长为 50 s,前 5 s 为系统初始化热身阶段,取后 45 s 的数据作为比较用数据。缓存替换算法采用默认的 LRU,对照组为 NDN 网络架构中常用的 5 种缓存放置策略 LCE、LCD、Betw、Probability 以及 ProbCache,其中,Probability 的缓存概率取值为 0.5。

3.2　性能指标

本文选用网络缓存研究中常用的 3 个指标对各策略进行衡量和比较:

1)全网命中率 *Ratio*。设兴趣包在服务器以外的节点中命中次数为 *hit*,兴趣包总数为 *total*,则 $Ratio=\frac{hit}{total}$。命中率能够直观反映缓存系统的优劣,命中率较高则说明缓存系统性能较好,其中有较大比例的请求都提前得到了响应,这正是缓存的重要作用之一。

2)服务器负载 *load*,为服务器每秒需要处理的兴趣包数量,该值越大,服务器越繁忙。服务器负载反映了服务器所承受的压力,如果服务器负载过高而服务器处理能力不足,将会造成极大的网络延迟甚至丢包。因此,在可能的情况下应尽量降低服务器负载。

3)平均时延 *delay*,反映了用户从发出请求到收到响应的平均时间,单位为 ms,该值越大,用户体验越差。缓存很重要的一个作用就是提升用户体验,因此,用户的请求时延是衡量缓存系统性能的一个重要指标。

3.3　各缓存策略性能对比与分析

本节通过改变节点缓存大小和 Zipf 参数来研究比较 3.2 节 3 个性能指标在各缓存策略下的表现。

3.3.1　全网命中率

图 2 所示为 6 种缓存策略下用户全网命中率随缓存大小的变化曲线,其中,Zipf 参数为 0.7。由图 2 可以看出,BEP 策略的全网命中率远高于 LCE 策略,当缓存容量为 0.025 时,BEP 的全网命中率相较 LCE 提高了 55.25% 。当节点缓存较小时,LCD 有比 Betw 更高的缓存命中率,当节点缓存较大时,Betw 的命中率高于 LCD,原因是缓存较小时,Betw 的缓存资源紧张,其高替换率的缺点变得尤其突出,导致其性能较差,但随着节点缓存大小的增加,单个节点能够缓存更多的内容,从而缓解了 Betw 的高替换率问题。而作为 Betw 的改进方案,BEP 在全网缓存命中率方面优于其他所有缓存放置策略,当缓存容量为 0.025 时,BEP 的全网缓存命中率相较 Betw 提高了 23.77% 。

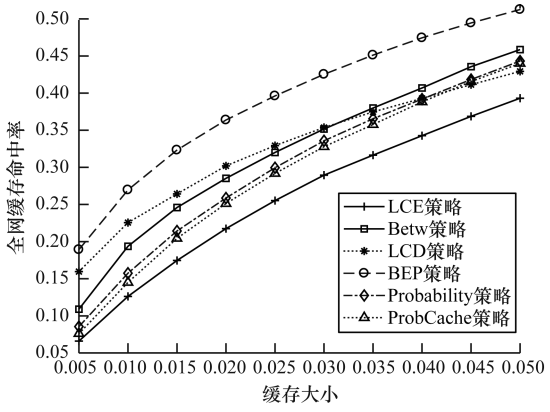

图 2　各策略全网缓存命中率随节点缓存大小的变化情况

将缓存容量固定为 0.025,改变 Zipf 参数时各缓存策略的全网缓存命中率变化情况如图 3 所示。从图 3可以看出,BEP 策略依然具有最优的性能。随着 Zipf 参数的增大,有更多的用户请求集中在更少的内容上,因此,当 Zipf 参数达到 1.5 时,大部分流行度高的请求都能够被缓存,6 种缓存策略的命中率都在 90% 左右,其中,BEP 策略高达 92.14% 。

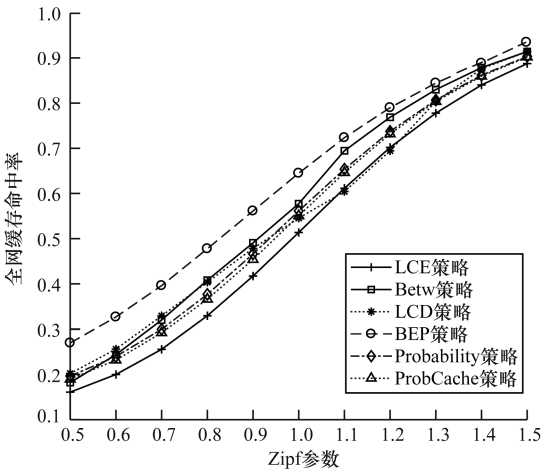

图 3 各策略全网缓存命中率随 Zipf 参数的变化情况

3.3.2 服务器负载

将 Zipf 参数固定为 0.7,改变节点缓存容量大小,各缓存策略下服务器负载变化情况如图 4 所示。从图 4 可以看出,随着节点缓存容量的增大,各策略的服务器负载呈现下降的趋势。将 Betw 与 LCD 进行对比,在缓存较小时,LCD 性能更优,而缓存较大时,Betw 性能更优。当缓存容量为 0.025 时,BEP 的服务器负载较 LCE、Betw、LCD 分别降低了 17.30% 、10.67% 、9.44% 。

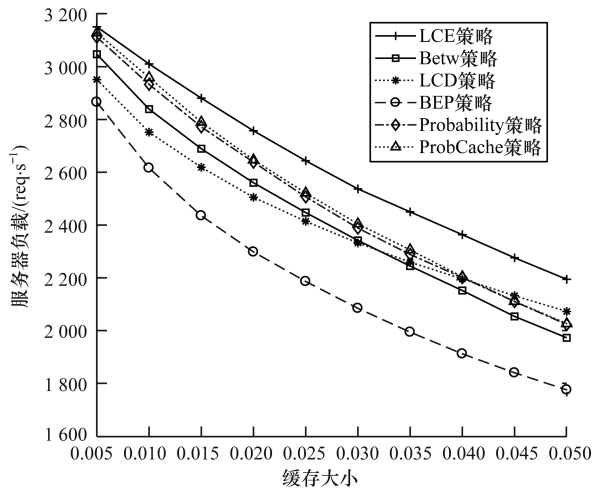

图 4 各策略服务器负载随节点缓存大小的变化情况

将缓存容量固定为 0.025,改变 Zipf 参数时各缓存策略的服务器负载变化情况如图 5 所示。从图 5可以看出,当 Zipf 参数足够大时,服务器负载可以达到相当低的水平,当 Zipf 参数为 1.5 时,BEP 服务器负载仅为 287.5 req/s,其他 5 种策略的服务器负载也在 420 req/s 以下,从整体来看,BEP 在服务器负载方面依然具有最优的性能。

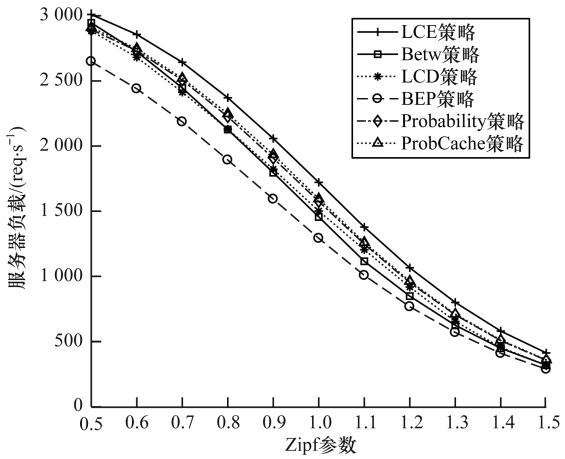

图 5 各策略服务器负载随 Zipf 参数的变化情况

3.3.3 平均时延

图 6 所示为 Zipf 参数为 0.7 时各缓存策略平均时延随节点缓存大小的变化情况,图 7 所示为节点缓存大小为 0.025 时各缓存策略平均时延随 Zipf 参数的变化情况。从中可以看出,本文 BEP 策略均优于其余 5 种策略,当 Zipf 参数为 0.7 且节点缓存大小为 0.025 时,BEP 平均时延比 LCE、Betw 和 LCD 分别降低 13.24% 、6.70% 和 4.43% 。

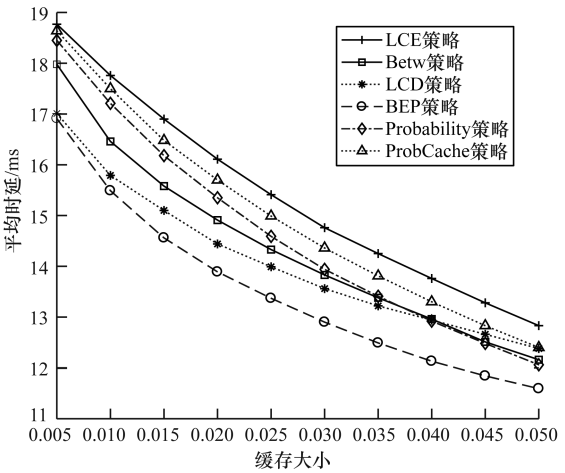

图 6 各策略平均时延随节点缓存大小的变化情况

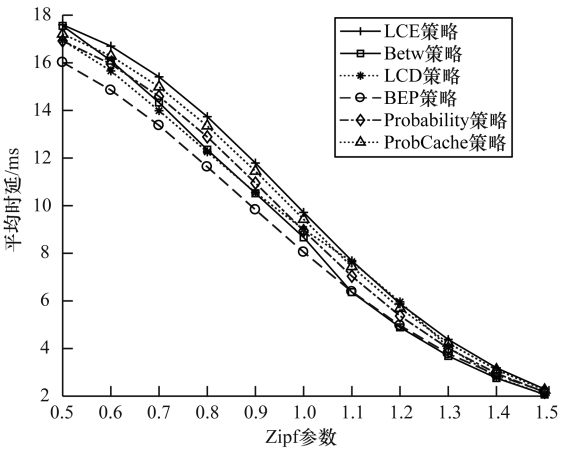

图 7 各策略平均时延随 Zipf 参数的变化情况

从图 7 还可以看出,当 Zipf 参数大于 1.1 时,本文 BEP 策略在平均时延上与 Betw 策略几乎相同,这是由于当 Zipf 参数较大时,更多的请求集中在更少的内容上,导致用户请求更加单一化,BEP 将最流行的内容放置在最重要的节点上,而 Betw 将所有内容都放置在最重要的节点上,当 Zipf 参数足够大时,几乎所有的用户请求都是流行的内容,因此两者性能相近。

4　结束语

本文结合节点的中心性、内容的流行度以及缓存对请求的过滤作用,将内容信息与拓扑信息相结合,提出一种缓存放置策略 BEP。该策略解决了 Betw 策略替换率过高的问题,同时避免了 LCE 策略的冗余与低效现象。仿真结果表明,相对 LCE、LCD 等策略,在全网缓存命中率和服务器负载方面,BEP 具有较好的性能表现。下一步考虑将缓存放置策略和路由机制相结合并利用更多的信息,以降低缓存成本,提升缓存收益,为互联网服务提供商或内容提供商呈现更友好的缓存策略。

参考文献

[1] VNI Cisco. Cisco visual networking index: forecast and methodology, 2016-2021[EB/OL]. [2018-01-02]. http://s2. q4cdn. com/230918913/files/doc _ downloads/report _ 2014/white_paper_c11-481360. pdf.

[2] ZHANG Lixia, AFANASYEV A, BURKE J, et al. Named data networking[J]. ACM SIGCOMM Computer Communication Review, 2014, 44(3): 66-73.

[3] AHLGREN B, DANNEWITZ C, IMBRENDA C, et al. A survey of information-centric networking[J]. IEEE Communications Magazine, 2012, 50(7): 26-36.

[4] JACOBSON V, SMETTERS D K, THORNTON J D, et al. Networking named content[C]//Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies. New York, USA: ACM Press, 2009: 1-12.

[5] CHAI W K, HE Diliang, PSARAS I, et al. Cache "less for more" in information-centric networks[J]. Computer Communications, 2013, 36(7): 758-770.

[6] 崔现东,刘江,黄韬,等. 基于节点介数和替换率的内容中心网络网内缓存策略[J]. 电子与信息学报, 2014, 36(1): 1-7.

[7] GILL A S, D'ACUNTO L, TRICHIAS K, et al. BidCache: auction-based in-network caching in ICN[C]//Proceedings of 2016 IEEE Globecom Workshops. Washington D. C., USA: IEEE Press, 2016: 1-6.

[8] LI Zhe, SIMON G. Time-shifted TV in content centric networks: the case for cooperative in-network caching[C]//Proceedings of IEEE International Conference on Communications. Washington D. C., USA: IEEE Press, 2011: 1-6.

[9] AOKI M, SHIGEYASU T. Effective content management technique based on cooperation cache among neighboring routers in content-centric networking[C]//Proceedings of International Conference on Advanced Information Networking and Applications Workshops. Washington D. C., USA: IEEE Press, 2017: 335-340.

[10] ZHANG Guoqiang, LI Yang, LIN Tao. Caching in information centric networking: a survey[J]. Computer Networks, 2013, 57(16): 3128-3141.

[11] LAOUTARIS N, CHE Hao, STAVRAKAKIS I. The LCD interconnection of LRU caches and its analysis[J]. Performance Evaluation, 2006, 63(7): 609-634.

[12] TARNOI S, SUKSOMBOON K, KUMWILAISAK W, et al. Performance of probabilistic caching and cache replacement policies for content-centric Networks[C]//Proceedings of 2014 IEEE Conference on Local Computer Networks. Washington D. C., USA: IEEE Press, 2014: 99-106.

[13] PSARAS I, CHAI W K, PAVLOU G. Probabilistic in-network caching for information-centric networks[C]//Proceedings of the 2nd Edition of the ICN Workshop on Information-centric Networking. New York, USA: ACM Press, 2012: 55-60.

[14] FREEMAN L C. A set of measures of centrality based on betweenness[J]. Sociometry, 1977, 40(1): 35-41.

[15] YU Meiju, LI Ru, LIU Yingqi, et al. A caching strategy based on content popularity and router level for NDN[C]//Proceedings of IEEE International Conference on Electronics Information and Emergency Communication. Washington D. C., USA: IEEE Press, 2017: 195-198.

[16] 丁尧,郑烇,郭晨,等. 基于节点热度与缓存替换率的 ICN 协作缓存[J]. 计算机工程, 2018, 44(2): 56-60, 67.

[17] 吴海博,李俊,智江. 基于概率的启发式 ICN 缓存内容放置方法[J]. 通信学报, 2016, 37(5): 62-72.

[18] CHE Hao, TUNG Y, WANG Zhijun. Hierarchical Web caching systems: modeling, design and experimental results[J]. IEEE Journal on Selected Areas in Communications, 2002, 20(7): 1305-1314.

[19] GUAN Jianfeng, QUAN Wei, XU Changqiao, et al. The location selection for CCN router based on the network centrality[C]//Proceedings of IEEE International Conference on Cloud Computing and Intelligent Systems. Washington D. C., USA: IEEE Press, 2013: 568-582.

[20] AFANASYEV A, MOISEENKO I, ZHANG Lixia. ndnSIM: NDN simulator for NS-3[EB/OL]. [2018-01-02]. http://www. named-data. net/techreport/TR005-ndnsim. pdf.

[21] MASTORAKIS S, AFANASYEV A, MOISEENKO I, et al. ndnSIM 2.0: a new version of the NDN simulator for NS-3[EB/OL]. [2018-01-02]. http://named-data. net/techreport/ndn-0028-1-ndnsim-v2. pdf.

[22] ZHANG Lixia. On the evolution of ndnSIM: an open-source simulator for NDN experi-mentation[M]. New York, USA: ACM Press, 2017.

[23] BRESLAU L, CAO Pei, FAN Li, et al. Web caching and Zipf-like distributions: evidence and implications[C]//Proceedings of the 18th Annual Joint Conference of the IEEE Computer and Communications Societies. Washington D. C., USA: IEEE Press, 1999: 126-134.

编辑　吴云芳