信息中心网络中基于局部内容活跃度的自适应缓存算法

田 铭 邬江兴 兰巨龙
(信息工程大学 郑州 450001)

摘 要 通过对信息中心网络的网内节点缓存建模,分析发现基于全局内容流行度的替换策略不适用于信息中心网络的分布式模式。继而提出了一种基于局部内容活跃度的缓存替换策略LAU,并基于该策略提出了一种自适应路径缓存算法ACAP,使缓存内容按照本地活跃度依次缓存在访问路径中。仿真结果表明,LAU策略提高了单节点缓存命中率;ACAP相比已有的路径缓存算法,具有较低的服务器命中率和跳数比。最后对该算法适用的缓存结构和拓扑结构进行了讨论和分析。

关键词 信息中心网络,缓存替换策略,缓存算法,内容活跃度
中图法分类号 TP393 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2016.11.032

Adaptive Cache Algorithm Based on Local Content Activity in Information-centric Networks

TIAN Ming WU Jiang-xing LAN Ju-long
(The PLA Information Engineering University, Zhengzhou 450001, China)

Abstract After modeling analysis, we found that replacement policy based on global content popularity does not suit for the distributed model of information centric networks. This paper presented a cache replacement policy based on local content activity called LAU. And an adaptive path caching algorithm named ACAP was proposed, so that the contents were cached successively in access path on the basis of local activity. The simulation results show that LAU strategy improves the cache hit rate of the single-node. Compared to the existing path caching algorithms, ACAP has low hit rate of server response and low hop ratio. At last, the applicable cache and topology structure of this algorithm were discussed and analyzed.

Keywords Information-centric networks, Cache replacement strategy, Cache algorithm, Content activity

1 引言

随着互联网技术与应用的飞速发展以及互联网用户的快速增长,宽带化、内容化与个性化已经成为互联网发展的主旋律。根据Cisco VNI Mobile Forecast统计分析和预测[1],全球IP流量在过去的5年中增长了5倍,在2014年—2019年间,网络流量仍将以23%的年均复合增长率高速增加,其中,大部分数据流量都源自于内容获取类应用。互联网2016年一年的数据流量将达到zettabyte(1ZB=270Byte),到2019年将超过2ZB,仅视频类业务流量就将占据所有IP流量的80%。

同时,随着人们对于数据内容的需求日益强烈,以用户驱动的内容分发类应用和数据传输将会呈现高速增长趋势,网络应用的主体逐步向内容获取和信息服务演进。互联网正历经从"以互联为中心"的主机通信到"以内容为中心"的演变历程。以信息为中心的网络(Information-Centric Network, ICN)[2-4]成为未来网络的一种重要模式和发展趋势。命名数据网络NDN[5]就是其中的代表,它利用沿途缓存(In-path Caching)方式,试图在网络各节点上用cache方式驻留高请求频度的服务内容信息,以便以最短传输路径尽可能地接近传送用户所需的内容数据。文献[6]指出,对于NDN网络,合理的内容放置和缓存决策是有效发挥网络性能的关键因素。但NDN泛滥式的沿途全部缓存方式(Cache Everything Everywhere, CE2)致使节点缓存内容趋于同质化,导致大量的缓存冗余。而且,在缓存决策时,缺乏对于内容本身差异化特征的考虑,无法实现内容的优化存储。为此,为了有效发挥内容普遍缓存的优势,高效缓存算法的设计就成为NDN需要解决的关键问题。

为了提高ICN的工作效率,提高缓存命中率,研究者就"单节点"和"路径"缓存展开了研究。

1.1 单节点内部缓存替换策略

当单节点内部需存储的内容超过缓存容量时,要采取一定的缓存替换策略,将"更有价值"的内容存储在本地。两种典型的缓存更新策略是LRU(Least Recently Used)[7]和

到稿日期:2015-10-29 返修日期:2016-01-19 本文受国家重点基础研究发展计划("973"计划)基金资助项目(2012CB315901, 2013CB329104),国家自然科学基金资助项目(61309019, 61372121),国家高技术研究发展计划("863"计划)基金资助项目(2015AA016102)资助。
田 铭(1984—),女,博士生,助理工程师,主要研究方向为信息中心网络、路由及缓存算法,E-mail:tianming19841101@126.com;邬江兴(1953—),男,教授,主要研究方向为通信与信息系统、计算机与网络技术;兰巨龙(1962—),男,博士,教授,主要研究方向为新型网络体系结构、路由与交换技术。

LFU(Least Frequently Used)[7]，它们分别对最近时间最少使用和长时间内最小频率使用的内容进行更新。然而 LRU 只能保存新近访问的内容，并未考虑内容的访问频度，对某个“非热点”内容的偶然访问也会引起对内容的存储，从而造成资源的浪费；LFU 策略对当前访问内容不敏感，过去时间内对某热点内容的大规模访问会一直在 LFU 中留下副本，直至当前访问频率超过该内容。朱轶等人[8]提出一种基于流行度的缓存概率置换策略，该策略根据请求数据块的流行度选择数据块在缓存队列中的置换位置，尽可能平衡不同流行度内容在网络中的分布。其通过数值计算表明了该策略的优良性能，但未通过实验仿真或者测试进行验证。

1.2 路径缓存节点选择策略

在数据包逐跳转发的过程中，经过的路由器节点可以存储该数据包。现有的路径缓存节点选择策略主要分为两类：CE2(Cache Everything Everywhere)[5] 和 CES(Cache Everything Selectively)[9-13]。

CE2 是将路径中转发的所有内容在经过的所有节点都缓存。每个节点内部执行缓存更新策略 LRU。在 NDN 协议中采用了 CE2 策略。这种策略简单易于执行，然而这种没有规划的泛滥式缓存使得路由器节点之间的缓存内容趋于一致，大量的缓存冗余造成了资源的浪费。

CES 是在路径传输过程中依策略挑选出某个节点存储数据内容。如 LCD(Leave Copy Down)[9] 是将缓存内容尽量放在离用户最近的边缘节点，降低缓存节点在网络中的层次；然而边缘节点的缓存容量总是有限的，大量的内容涌向边缘节点，使得内部的更新策略极为频繁，高层次的节点缓存又得不到有效利用。ProbCache(Probabilistic Caching)[10] 是在评估路径缓存容量的基础上，联合考虑 LCD 思想，将节点在路径中的层次作为权重系数，计算路径中每个节点的缓存概率，并依概率缓存。仿真表明，该策略在节省跳数和降低服务器命中率方面较 CE2、LCD 有更优的性能。文献[11]指出在网络拓扑中挑选连接度较高的节点存储数据内容，这样可满足更多下游节点的内容请求。文献[12]表明参考流量特性选择在网络中的缓存位置，VoD 内容应该尽量缓存在网络边缘，而其它流量类型的内容应尽量存储在网络核心。文献[13]提出了一种对等自治系统之间的缓存协调模型，重点考虑了是否通过显式的通告达到所有内容的可用性。

刘外喜等人[14]提出了把内容的放置、发现、替换统一起来考虑的 APDR 机制，实现内容的有序缓存。APDR 的主要思想是：兴趣报文除了携带对内容的请求，还收集沿途各节点对该内容的潜在需求、空闲缓存等信息，使得兴趣包的汇聚点和目的地节点可以据此计算出一个缓存方案，并把该方案附加在数据报文之上，通知返程途中的某些节点缓存该内容并设置指定的缓存时间。另外，文献[15]提出了选择性缓存机制，采用带宽换缓存的思想，利用链路的冗余带宽，将内容分流到相邻节点缓存，从而有效提升缓存效率，但是需要付出较大的额外开销。文献[16]提出了一种基于节点介数的缓存策略(betw)，请求内容只存储在传输路径中介数最大的节点，但该方案会引起介数较大节点严重的缓存过载，而其它节点缓存又得不到有效利用。文献[17]采用“差异化缓存”的方式，提出一种依据内容请求序列相关性的协作缓存算法，该算法根据内容活跃度的变化趋势，空间维度上逐跳推进内容存储位置，时间维度上动态调整内容缓存时间，以渐进式的方式将流行的请求内容推送至网络边缘存储。但是，逐跳推进内容存储位置容易引起热点内容的“慢响应”，而且动态差异化地调整缓存驻留时间也引入了额外的计算、存储和计时开销。文献[18]从信息熵的角度提出隐私度量指标，以增大攻击者的不确定度为目标，对于缓存策略的合理性给予了证明；其次，通过构建空间匿名区域，扩大了用户匿名集合，增大了缓存内容的归属不确定性。缓存决策时，针对垂直请求路径和水平匿名区域，分别提出沿途热点缓存和局域 hash 协同的存储策略，减小缓存冗余和隐私信息泄漏。但是，文中匿名区域采用集中式的管理方式，管理节点维护大量的用户访问和内容存储信息，使得该节点成为隐私保护的薄弱环节。文献[19]提出一种缓存空间的动态借调机制，将相对空间的缓存资源分配给需求程度更大的节点支配，以换取过载节点性能的提升。其本质是利用其它邻居节点存储本身被替换的内容信息，并在本节点保留该内容的索引。该方案仅限于在邻居之间进行，限制了应用的范围。文献[20]提出以社团为单位，把内容缓存到社团内用户容易获取的节点，并且提出一种节点混合替换策略，使不同流行度的内容对象在各社团内部的时间分布上更为合理。但是，该策略中社团内部节点重要度低的节点采用的替换策略使得较为活跃的内容被替换掉，无形中造成了缓存命中率低的缺点。文献[21]提出了以文件为单位的协作缓存策略，为每个文件挑选一个跟踪节点指向所有文件块，这种细粒度的协作缓存方式会引入较大的信令开销。张国强等[22,23]在综述性论文中也详细总结和分析了信息中心网络的缓存策略。

1.3 缓存策略分析

上述策略多是从选择路径中的缓存节点出发，并未与节点内部的缓存替换策略有效结合，使更多“热点”内容分布在网络各个节点(边缘和核心)，故算法性能仍有待提升。

本文提出“节点-路径”联合内容缓存策略，“节点”即在单个网络缓存节点用内容活跃度来综合评价内容的“热度”和“新颖度”，提出最小活跃度(Actively)替换策略 LAU(Least Actively Used)；“路径”即在内容传输路径上依照自适应缓存策略，将数据内容依照内容活跃度安排在不同层次的节点缓存位置上。联合缓存策略的目的即在整个内容通信过程中合理安排节点和路径缓存策略，将更活跃的内容推向网络边缘存储，并充分利用网络核心节点的存储资源，提高缓存资源利用率和边缘 Cache 命中率(hit rate)，缩短响应距离，减少服务器响应率。

本文第 2 节分析信息中心网络的缓存模型，指出局部内容流行度的必要性；第 3 节定义内容活跃度和相应的数据统计结构；第 4 节提出基于内容活跃度的单节点和路径缓存算法，并给出复杂度分析；第 5 节进行仿真实验；最后总结全文。

2 信息中心网络的缓存模型

为清晰描述 ICN 缓存共享问题模型，首先给出如下假定。

网络拓扑。ICN 的服务器与请求内容的用户之间构成了树形拓扑，为了分析方便，考虑单个服务器构成的二叉树形拓

扑，如图1所示。二叉树拓扑中，服务器为根节点，其余 M 层为路由节点，第 i 层节点的缓存容量为 x_i，$i=1,2,\cdots,M$。

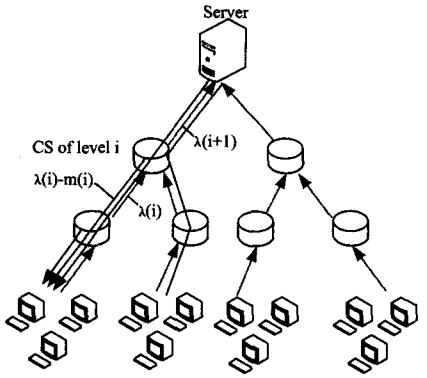

图1 二叉树形拓扑

网络中内容数量和分布。令 $F=\{f_1,f_2,\cdots,f_{|F|}\}$ 表示ICN中内容对象的集合，$|F|$ 为内容对象的数量。假定内容对象大小相同，即 $S=S_1=S_2=\cdots=S_{|F|}$，不妨令 $S=1$。内容流行度分布为 $\{q_k,k=1,2,\cdots,|F|\}$，并且有：

$$q_k=\frac{c}{k^\alpha},c=(\sum_{k=1}^{|F|}\frac{1}{k^\alpha})^{-1}\text{且}\ \alpha>1 \tag{1}$$

请求到达过程。请求到达为MMRP(Markov Modulated Rate Process)[24]过程模型，即对 F 中第 k 个内容的请求以泊松过程产生，并且其请求到达速率满足 $\lambda_k=\lambda q_k$。

树形拓扑的第 i 层节点上请求到达速率为 $\lambda(i)$，请求缺失速率为 $m(i)$。$p_k(i)$ 为节点 i 上第 k 个内容的缓存缺失概率，$q(i)$ 为节点 i 上的内容流行度分布。对于MMRP到达过程，树形拓扑的第 i 层节点的内容缺失速率可以近似为：

$$m(i)=\sum_{k=1}^{|F|}p_k(i)\lambda_k(i),i\geqslant 1 \tag{2}$$

第 i 层节点的内容流行度分布 $\{q_k,k=1,2,\cdots,|F|\}$ 满足：

$$\begin{aligned}q_k(i)&=\frac{\lambda_k(i)}{\sum_{l=1}^{|F|}\lambda_l(i)}=\frac{q_kp_k(1)p_k(2)\cdots p_k(i-1)}{\sum_{l=1}^{|F|}q_lp_l(1)p_l(2)\cdots p_l(i-1)}\\&=\frac{\prod_{j=1}^{i-1}p_k(j)q_k}{\sum_{l=1}^{|F|}\prod_{j=1}^{i-1}p_l(j)q_l},i\geqslant 1\end{aligned} \tag{3}$$

对于图1中的二叉树形拓扑，第 i 层节点上的请求到达速率等于其子节点的内容缺失速率之和，即：

$$\lambda(i)\equiv 2m(i-1)=2\sum_{k=1}^{|F|}p_k(i-1)\lambda_k(i-1),i\geqslant 1 \tag{4}$$

由式(3)可以看出，节点 i 上的到达请求是由其它节点向上转发的缺失请求汇聚而成的，并且其流行度分布不符合初始的用户请求流行度分布 $\{q_k(i),k=1,2,\cdots,|F|\}$。在实际的ICN网络中，用户请求和流行度分布是动态变化的，各层节点上内容请求的分布具有差异性。为此，缓存算法需要根据本地用户请求模式来决定如何部署内容，而不能根据全局的内容流行度统一部署缓存策略。故本文提出内容活跃度的概念，来刻画内容对象被本地用户请求的频繁程度和时间上的临近性。

3 内容活跃度

通过上述对LRU和LFU策略的分析可以得出，缓存替换策略必须满足：1)能够适应快速变化动态网络传输模式；2)能够不受临时性的内容交替影响。

为了满足这两点需求，本文提出一种最小活跃度(Activity)替换策略LAU(Least Actively Used)，将内容的活跃度因素引入缓存替换算法中，其主要特征是：1)为了使缓存内容更好地匹配内容访问的动态性特征，引入“滑动窗口”的概念；2)只有落入窗口内的内容请求才被统计计算内容活跃度，对于“陈旧”的历史信息通过窗口的滑动使之落在统计窗口之外；3)对窗口内部的内容请求又分别赋予不同的权值，越“新颖”的内容对于活跃度的贡献越大。

3.1 定义

表1列出了内容活跃度定义中涉及的变量符号涵义。

表1 内容活跃度涉及的变量及含义

变量	含义
$\|F\|$	网络中所有内容的总数
T	滑动窗口的最小滑动间隔
K·T	滑动窗口的总宽度。
[0,(K−1)·T]	内容请求的历史参考窗口
(K−1)*T,[(K−1)·T,K·T]	内容请求的当前观测窗口
$V_{i,j}$	时间区间[(j−1)×T,j×T] 内容i的请求次数
$F_{i,j}$	时间区间[(j−1)×T,j×T]内容i的请求频率
$V_i(t)$	时间区间[(K−1)×T,t]内容i的请求次数
$F_i(t)$	时间区间[(K−1)×T,t]内容i的请求频率
α	权值参数
$A_i(t)$	截止到当前时刻t内容i的活跃度

假定当前时刻为 t，以时间间隔 T 对内容对象 i($1\leqslant i\leqslant|F|$)的访问次数进行统计，记 $V_{i,1},V_{i,2},\cdots,V_{i,n-1},V_{i,n},V_i(t)$ 分别表示时间区间 $[0,T),[T,2T),\cdots,[(n-1)T,nT),[(n-1)T,t)$ 内的用户请求次数。当 $t=nT$ 时，$V_i(t)=V_{i,n}$。

时间间隔 T 内的内容请求频率为 $F_{i,j}=V_{i,j}/T$，其中 $j=1,2,\cdots,K$ 并且 $F_i(t)=V_i(t)/[t-(K-1)T]$ 表示滑动窗口最新间隔内的请求频率。

内容活跃度定义为内容对象 i 在与当前时刻 t 相关的滑动窗口内的加权请求频率。具体可以表示为：

$$A_i(t)=\sum_{j=1}^{K-1}\frac{F_{i,n-K+j}}{\alpha^{K-j}}+F_i(t) \tag{5}$$

其中，α 为权值参数，且 $\alpha>1$，K 为滑动窗口的宽度参数，即所取的时间区间 T 的个数。

内容活跃度的动态更新是用户请求驱动的，只有当新的请求到达时，才对内容对象的活跃度进行更新。滑动窗口的概念及滑动间隔示意图如图2所示。

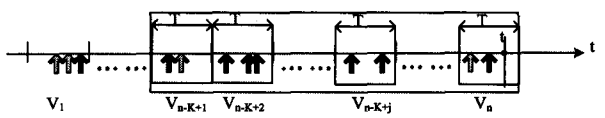

图2 内容活跃度定义示意图

当 $\alpha\to 1$ 时，$A_i(t)$ 为滑动窗口内所有内容的访问频率，若窗口宽度无限大时，LAU策略与LFU策略效果相同。当 $\alpha\to\infty$ 时，历史参考窗口内的访问频率随 α 的指数增长而快速衰减，$A_i(t)$ 主要反映了当前观测窗口内的内容请求频率，LAU策略与LRU策略效果相近。

内容活跃度与内容的流行度相比，具有不同的含义。活跃度具有局部意义，即只对特定的存储节点有意义，即单个缓存节点上内容的分布信息。而内容的流行度(popularity)反

映的是内容分发系统中单个内容相对于其它内容的流行程度，具有全局意义。

3.2 参数设置

为了更加准确地描绘内容访问的频率随时间的变化关系，即内容频繁访问的时间尺度，使内容活跃度更加贴切地反映某段时间内的内容相对访问频率，需要对滑动窗口的滑动间隔和窗口宽度做出合理设置。

文献[25]指出，对文件下载日志和流媒体访问日志的分析表明，多数流行度较高的文件都会在发布之后的若干小时内达到流行度的高峰。文中分析了来自两个实际系统的访问日志，一个是多媒体文件共享 FTP 服务器，另一个是 PowerInfo VoD 视频点播系统[26]。从这两个日志中文件的流行度的时间序列分析得到两种流行度变化的模式。

(1)模式 A(Pattern A)。代表了一类在发布初期就变得很流行的文件，比如新的电影或音乐专辑，但在发布初期的一阵高峰后会慢慢趋于平静。

(2)模式 B(Pattern B)。在发布的初期并没有突发的访问量高峰。它只是以 1 天为周期稳定地振动。模式 B 的曲线和模式 A 稳定后的曲线一致，它代表了不热门的文件。

结合滑动窗口的设计目标，综合上述模式分析发现：

(1)对于模式 A 的内容访问，要在内容活跃度的上升期将之缓存，滑动窗口宽度不宜超过文件到达高峰期所需的时间，这里设 $KT=1$ 小时；滑动窗口的滑动间隔要精细到判别内容访问的急速上升和下降，越精细越好。

(2)对于模式 B 的内容访问，由于一直处于稳定振动的低活跃度状态，对滑动窗口的滑动间隔并不敏感，因此滑动间隔不宜设置得过于精细，否则频繁统计内容活跃度将造成资源浪费。综上所述，滑动窗口的滑动间隔设为分钟级，如 $T=1$ 分钟。

3.3 一种基于光谱布鲁姆过滤器的活跃度统计方法

随着内容请求数量和频次的增加，数据集合变得越来越大，集合元素的表示和访问就愈加困难，如何表示内容请求信息，并实现在海量信息中快速、高效地查找所需资源，成为活跃度统计的基础。

作为一种精简的信息表示方案，布鲁姆过滤器(Bloom filter)能够满足高速网络发展中的高效资源交互和查找需求。然而标准布鲁姆过滤器只支持插入和查询操作，文献[27]提出光谱布鲁姆过滤器。假设集合的元素具有多样性(multiplicity)，即集合中的元素可能存在多个副本。用元素对应的 k 个 Counter 计数器序列中最小的计数器作为元素出现的频度估计，从而可对元素的出现频度做出比较。具体系统结构如图 3 所示。

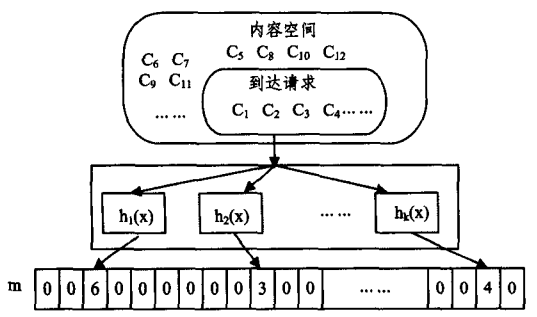

图 3 内容访问频次的统计数据结构

对每个内容条目请求次数的统计是针对集合中的元素进行频度统计。首先对光谱布鲁姆过滤器初始化，即将计数器向量的各位初始化为 0。集合中元素插入过程就是元素到向量的映射过程。当有内容请求 C_i 到达路由节点时，计算元素 C_i 的 k 个哈希地址，即 $h_1(C_i),h_2(C_i),\cdots,h_k(C_i)$；将对应位置的计数器读数执行加 1 操作，即完成了内容请求的计数操作。当需要读取内容请求 C_i 的出现次数时，首先由哈希函数计算哈希地址 $h_1(C_i),h_2(C_i),\cdots,h_k(C_i)$；设各计数器读数分别为 $BF[h_1(C_i)],BF[h_2(C_i)],\cdots,BF[h_k(C_i)]$，那么内容 C_i 的请求次数为 $V_i=\min\{BF[h_1(C_i)],BF[h_2(C_i)],\cdots,BF[h_k(C_i)]\}$。

为实现不同时间间隔请求频率的加权运算，滑动窗口内包含的每个时间间隔对应一个 SBF，用于统计本次时间间隔内每个内容 i 的请求次数 $V_{i,n}$，再由活跃度定义，由多个 SBF 统计读数的加权平均计算得到。

$$A_i(t)=\sum_{j=1}^{K-1}\frac{V_{i,n-K+j}}{T\cdot\alpha^{K-j}}+\frac{V_i(t)}{t-(K-1)\cdot T} \tag{6}$$

对于每次滑动窗口的滑动操作，SBF 不需要执行删除运算，只需要将对应过期时间间隔的 SBF 读数清零，用以继续统计新的观测窗口的内容请求即可。

4 基于内容活跃度的自适应缓存算法

4.1 单节点的最小活跃度替换策略

内容活跃度考虑了内容请求在时间和地域上的临近性，并且通过滑动窗口机制，自适应地反映用户请求的变化，而缓存替换策略的目标就是要最大程度地匹配用户的请求。基于此，本文提出一种面向单节点的最小活跃度替换策略，将收到的内容与节点已缓存内容条目的活跃度大小做比较，替换滑动窗口内活跃度最低的内容条目。主要的算法流程如算法 1 所示。

算法 1 最小活跃度替换策略 LAU

Step1 对于新到达的数据内容 C_i，记录数据内容和到达时刻 t，并判断滑动窗口是否满足滑动条件：$t-nT>T$，如果满足，则 $n\leftarrow n+1$，窗口向前滑动 T；否则，执行 Step2；

Step2 根据访问历史记录计算 t 时刻对 C_i 所请求内容的活跃度 $A_i(t)$和已缓存内容的活跃度$\{A_j(t),j=1,2,\cdots,J\}$；

Step3 如果 $A_i(t)>\min_{j\in J}\{A_j(t)\}$，执行 Step4；否则，执行 Step5；

Step4 将 C_i 替换掉缓存中活跃度最低的内容对象，执行 Step6；

Step5 丢弃 C_i；

Step6 算法结束。

4.2 一种基于 LAU 的路径自适应缓存算法

ICN 网络中多个缓存节点构成了缓存群组，由于路径中各个节点用户请求存在一定的相似性，每个节点都针对自身的访问请求部署内容，导致所存储的内容趋同。为了避免这种情况，缓存在决定是否存储一个内容时，需要考虑该内容在本地的活跃度以及在其它缓存上是否已经有此内容的副本，以“协调”的方式提供整体上更高的缓存命中率。然而，在已有的研究中，为了实现多个缓存节点的内容协调，多以“显式”的方式探测路径的缓存容量，增加了传输代价，复杂度较高，难以在实际网络中部署应用。

本文提出一种基于最小活跃度替换策略(LAU)的路径

自适应缓存算法(Adaptive Caching Algorithm for Path, ACAP),充分地结合了 ICN 网络中的逐跳路由模式和将兴趣包汇聚的 PIT 表项特性,以"隐式"的协调方式完成了内容在路径中的自适应缓存部署。主要特征包括:

(1)基于最小活跃度替换策略,而非 LRU 策略,存储经由每个节点转发的在本地是较为活跃的内容,克服了 LRU 策略导致的频繁更替现象。

(2)存储了某活跃内容的下游节点不会将请求转发至上游节点,故而上游节点增加了存储其它活跃内容的概率,克服 CE2 策略导致的缓存趋同现象。

(3)以 Bloom Filter 的数据结构统计本地缓存空间中的内容活跃度,以较低的存储空间代价和计算复杂度支持了内容活跃度的高效统计和快速查询。

具体的算法如算法 2 所示。

算法 2 路径自适应缓存算法 ACAP

Step1 初始化滑动窗口和缓存队列 $Q_k(1\leqslant k\leqslant J)$,n 为滑动窗口的起始时间区间,J 为缓存节点上存储的内容对象的个数。

Step2 对于新到达请求 q_i,判断所请求内容是否在本地缓存,如果存在,将 q_i 所请求内容在请求端口做出响应,执行 Step7。否则,执行 Step3。

Step3 判断所请求内容在 PIT 表项中是否存在,若存在,将当前请求对应端口加入到 PIT 表项中,执行 Step5;否则,执行 Step4。

Step4 查找 FIB 表请求内容对应的下一跳转发端口,转发请求并在 PIT 表中记录当前请求对应的内容和请求端口。

Step5 等待输入端口的数据内容回复情况。若有新到达数据内容 C_i,执行算法 1 最小活跃度替换策略 LAU,否则执行 Step5。

Step6 查找 PIT 表中数据内容对应的 q_i 请求端口转发数据。

Step7 算法结束。

4.3 算法复杂度分析

结论 1 若节点中已缓存的内容条目为 J,则 LAU 的时间复杂度为 $O(J)$。

LAU 实现过程中,对于每一个到达的内容请求,执行一次活跃度的比较运算,每次运算处理中循环的次数为该节点缓存的内容数目 J,同时可能包含对最低活跃度内容的替换。最坏情况下,执行次数为 $J+1$ 次。所以 LAU 的时间复杂度为 $O(J+1)\approx O(J)$。

结论 2 若滑动窗口的宽度为 KT,布鲁姆过滤器的位向量长度为 m,则 LAU 的空间复杂度为 $S(K\cdot m)$。

LAU 算法执行过程中,设置滑动窗口来记录不同的时间间隔,每个时间间隔内通过光谱型布鲁姆过滤器记录到达的内容请求及相应的频次。滑动窗口宽度为 KT 时,布鲁姆过滤器的个数为 K 个;每个过滤器将个数为 N 的内容条目映射到长度为 m 的位向量中,作为计算内容活跃度信息的依据。所以,LAU 的空间复杂度为 $S(K\cdot m)$。

结论 3 若节点中已缓存的内容条目为 J,总的内容条目为 $|F|$,则 ACAP 算法的时间复杂度为 $O(3|F|+|F|\cdot J-J^2-J)$。

ACAP 算法在本地缓存中搜索请求内容的过程的时间复杂度为 $O(J)$;若搜索失败,则检查 PIT 表项,PIT 表项的最大规模为 $|F|-J$,时间复杂度为 $O(|F|-J)$;若 PIT 表项未匹配,则查找 FIB 表项,该表的最大规模为 $|F|$,时间复杂度为 $O(|F|)$;转发请求包后,等待响应数据包,等待新数据到达的最差情况是第 $|F|-J$ 个数据才是该请求的响应,执行 LAU 算法的复杂度为 $O(J)$,那么时间复杂度为 $O[(|F|-J)\cdot J]$;最后再次查找 PIT 表项的复杂度为 $O(|F|-J)$。那么总的复杂度为 $O[J+(|F|-J)+|F|+(|F|-J)J+|F|-J]=O(3|F|+|F|\cdot J-J^2-J)$。

结论 4 若滑动窗口的宽度为 KT,布鲁姆过滤器的位向量长度为 m,则 ACAP 的空间复杂度为 $S(K\cdot m)$。

ACAP 算法引入的空间存储代价与 LAU 相同,节点间不需要控制通信类的消息,只需节点内部统计内容活跃度即可,故 ACAP 的空间复杂度为 $S(K\cdot m)$。

5 仿真分析

5.1 仿真环境与性能指标

为了验证单节点 LAU 策略和路径自适应缓存算法 ACAP 的性能,使用 MATLAB 工具进行仿真实验。内容流行度服从 Zipf-Mandelbrot 分布[28],整形参数 $\sigma=0.7$,移动参数 $q=10$。系统中包含的内容个数 $|F|=1000$。仿真实验的硬件环境如下:处理器主频为 2.0GHz,512M 内存,Windows XP 操作系统。

为评估 LAU 策略和 ACAP 算法的性能,引入以下 3 个性能参数。

(1)缓存命中率(Cache Hit Ratio,CHR)

指路由节点在一定时间内,本地缓存中的内容做出的响应次数与其接收到的请求次数的比例关系。CHR 越大,说明网络中重复请求数目减少得越多,随之冗余流量减少得越多,缓存管理策略越有效。

(2)响应跳数比(Response Hop Ratio,RHR)

表示实际请求响应经过的跳数与在服务器端响应的跳数之比率。响应跳数比越大,请求得到响应的距离越长。当 $RHR=1$ 时,请求在所有节点的缓存中未得到响应,转发到服务器端响应。平均 RHR 越小,路径缓存算法越有效。

(3)服务器命中率(ServerHit Ratio,SHR)

指用户发送的内容请求在服务器端的响应次数与总的内容请求次数的比例关系。服务器命中率越小,内容请求的响应在路径缓存中得到响应的比例越高,经过的跳数越少,网络消耗越少。

5.2 单缓存命中率仿真

为量化基于内容活跃度的缓存替换策略的性能优势,以缓存命中率为指标进行仿真实验,取 $|F|=1000$。为了使 LAU 策略发挥"新颖度"和"热度"的优势,首先考察权值参数 α 的最佳取值。图 4 给出了权值参数 $\alpha=1,2,10,100$ 时 LAU 策略的命中率随缓存容量的变化曲线。就实验中所取的数值来看,权值参数 $\alpha=2$ 时,缓存命中率最高;$\alpha=1$ 及 $\alpha=10$ 时,缓存命中率比 $\alpha=2$ 时略低;当 $\alpha=100$ 时,缓存命中率最低。可见,α 的最佳取值在 1 和 10 之间,此时既考虑了历史参考窗口内的内容请求,又不夸大对当前观测窗口内容请求的影响,缓存命中率较高。实际中需要根据用户请求到达过程选择合适的算法参数。

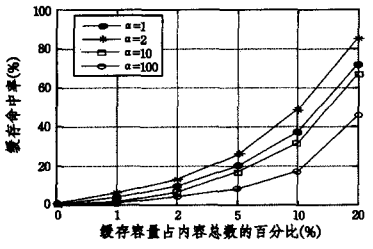

图 4　不同权值参数的缓存命中率

图 5 给出了单节点缓存替换策略的性能差异。目标算法为 Random 替换策略、LRU 策略、LFU 策略、本文提出的 LAU 策略，取 $\alpha=2$。Random 策略的命中率一直保持在较低水平，因为随机替换策略很难保证热点信息被缓存。LFU 策略高于 LRU 策略，LAU 算法的命中率要高于 LFU 和 LRU 算法，由于 LRU 算法只考虑请求到达的时间临近性，LFU 算法只考虑访问频率，而 LAU 综合考虑了两种因素，在缓存准入和替换上更为精确。

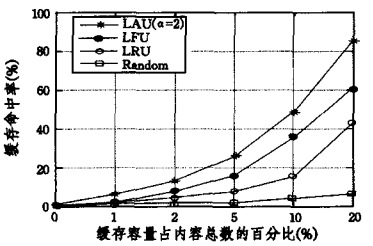

图 5　单节点缓存算法的命中率

通过对图 4、图 5 的对比发现，$\alpha=1$ 时的命中率和 LFU 策略相近，$\alpha=100$ 时的命中率和 LRU 策略相近。而 $\alpha=1$ 和 $\alpha=100$ 的命中率稍高于 LFU 和 LRU 策略，因为 $\alpha=1$ 时的活跃度反映的是滑动窗口内的访问频率，又过滤掉了 LFU 中陈旧的历史信息；$\alpha=100$ 时的活跃度主要反映的是滑动窗口的当前观测窗口内的访问频率，不同于 LRU 只考虑最近一次访问的内容信息，故而命中率性能较好。

5.3　路径自适应缓存算法仿真

首先采用层次化结构，仿真环境采用文献[10]中的拓扑设置，描述如下：网络拓扑为一个 6 层的二叉树拓扑，总的节点数为 127，根节点代表服务器。假定用户与骨干网路由器没有直接相连，请求从树最底部的两层节点产生，共产生 10000 个请求。

5.3.1　缓存容量对性能的影响

为检验缓存容量对算法的性能影响，缓存设置分为两种情况进行仿真验证。

(1)各节点缓存容量相同

分别取各节点的缓存容量为缓存内容总数的 1%，2%，5%，10%，20%，验证各算法的服务器命中率、跳数比等指标。路径缓存策略的目标算法为：1)沿途缓存策略(CE2)[5]，并且缓存替换采用 LRU，记为 CE2+LRU；2)文献[9]提出的 LCD 缓存方法，缓存替换采用 LRU，记为 LCD+LRU；3)概率缓存，从服务器到请求者的路径沿途所有节点以 0.7 的概率缓存内容，记为 P(0.7)+LRU；4)概率缓存，从服务器到请求者的路径沿途所有节点以 0.3 的概率缓存内容，记为P(0.3)+LRU；5)本文提出的自适应缓存算法 ACAP。

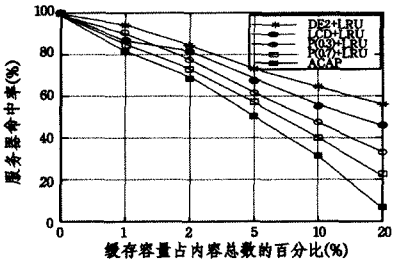

图 6　服务器命中率随缓存容量的变化曲线

由图 6 可以看出，路径自适应缓存算法相对于沿途缓存策略在服务器命中率上有很大改善，尤其是当缓存容量占内容总数的 20%左右时，服务器响应命中率减少到 10%左右，极大地降低了服务器负载，减少了网络传输代价。由于 CE2 策略使得路径中节点缓存内容趋同，因此随着缓存容量增加，服务器响应的命中率并未得到显著改善。相对于其它的缓存策略，ACAP 的服务器命中率有不同程度的改善，从而验证了该算法的有效性。

由图 7 可以看出，各个缓存算法随着缓存容量增加，平均访问跳数都有不同程度的减少。就 LCD 算法来说，虽然在服务器命中率上要高于 P(0.3)+LRU 算法和 P(0.7)+LRU 算法，但是 LCD 算法主要考虑将内容推向离用户最近的缓存节点，其平均访问跳数要小于上述两种算法。而本文提出的 ACAP 算法在缓存容量增加到内容总数的 20%时，跳数比减少到 57%左右，即用户发出的内容请求基本上只经过距离服务器 57%的跳数就可以得到响应。

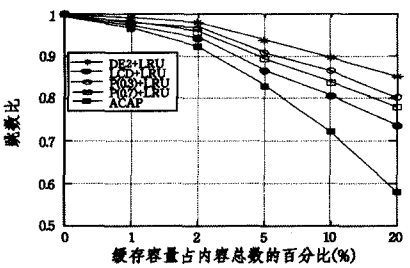

图 7　跳数比随缓存容量的变化曲线

(2)缓存容量随节点的层次发生变化

目标算法为：1)缓存容量从边缘到核心逐渐增大(Edge→core，简写为 E→c)，设核心根节点层次为 1，那么从 1—6 层的缓存容量分别为 10%，8%，6%，4%，2%，0；2)缓存容量从边缘到核心逐渐减小(Core→edge，简写为 C→e)，设核心根节点层次为 1，那么从 1—6 层的缓存容量分别为 0，2%，4%，6%，8%，10%；3)缓存容量保持不变，各层次的缓存容量都为内容数目的 5%。

图 8 给出了缓存容量变化与不变时，各层次缓存命中率的变化曲线。由图中可以看出，缓存容量固定为 5%时，各层次的缓存命中率基本保持在 25%左右，这与图 5 中单缓存替换策略 LAU 的命中率一致。在边缘节点设置较大缓存时(E→c)，边缘缓存的命中率为 50%左右，核心节点缓存容量较少，命中率也逐渐降低。而 ACAP(C→e)则与之相反，核心节点比其它层次节点的命中率较高，但仍低于 50%，这是因为靠近边缘的各层节点分别响应了部分内容请求，使得缓存容量最大的核心节点命中率低于 ACAP(E→c)的边缘节点。

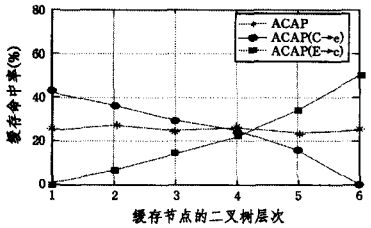

图 8　缓存异质与同质时不同层次节点的缓存命中率

图 9 给出了随着平均缓存容量增加，缓存容量异质与同质时，服务器命中率的变化曲线。由图中可以看出，缓存容量同质与边缘缓存较大时，服务器命中率较为接近，在小缓存容量时，边缘缓存较大的 ACAP(E→c)的服务器命中率更低。而核心缓存容量较大的 ACAP(C→e)服务器命中率一直高于缓存同质的 ACAP 和缓存异质的 ACAP(E→c)。

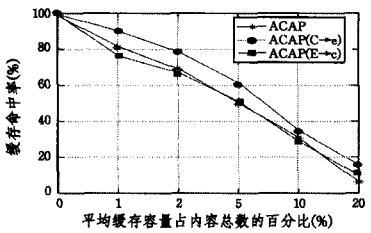

图 9　异质缓存时的服务器命中率变化曲线

图 10 给出了随着平均缓存容量增加，缓存容量异质与同质时，平均响应跳数比的变化曲线。由图 10 可以看出，边缘缓存较大的 ACAP(E→c)跳数比性能最优，缓存容量同质的 ACAP 性能次之，而核心缓存较大的 ACAP(C→e)跳数比最大。因为边缘缓存越大，用户响应的距离越短。由图 9 和图 10 可见，无论是服务器命中率还是跳数比性能，路径自适应缓存算法不适用于在核心节点部署较大缓存。

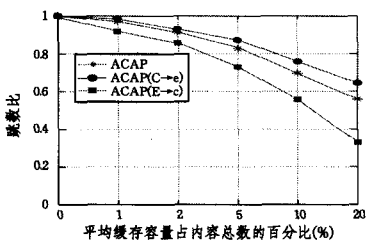

图 10　缓存异质与同质时跳数比的变化曲线

5.3.2　拓扑结构对性能的影响

为检验算法 ACAP 对网络拓扑的敏感性，分别用网络拓扑产生器给出 3 种拓扑结构：1)层次化结构(hierachical)。仿真环境采用文献[11]中的拓扑设置，描述如下：网络拓扑为一个 6 层的二叉树拓扑，总的节点数为 127，根节点代表服务器。假定用户与骨干网路由器没有直接相连，请求从树最底部的两层节点产生，共 96 个用户产生 10000 个请求。2)随机拓扑(Random)。其中节点数目为 127 个，节点平均度数为 3，服务器位置随机选择与网络中的某路由节点相连，用户数目为 96 个，请求数目为 10000 个。3)幂律结构(Power-law)。网络节点数目为 127 个，节点度数服从幂律分布。服务器位置随机选择与网络中的某路由节点相连，用户数目为 96 个，请求数目为 10000 个。

图 11 给出了不同拓扑结构下，服务器命中率随着缓存容量增加的变化曲线。由图 11 可以看出，无论缓存容量大小，层次化结构的服务器命中率一直处于最低水平。在大缓存容量时，幂律拓扑结构下的服务器命中率和层次化拓扑结构的命中率性能相近。而在随机拓扑结构下，服务器命中率较大，这是因为提出的 ACAP 算法适合于层次化结构的请求汇聚特性，随机拓扑下，网络节点连接关系的汇聚特性最差，所以服务器命中率最高。

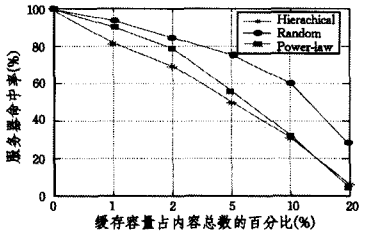

图 11　不同拓扑结构下的服务器命中率变化曲线

图 12 给出了不同拓扑结构下，跳数比随着缓存容量增加的变化曲线。在小缓存容量时，幂律拓扑结构下的跳数比和随机拓扑结构的跳数比性能相近。而随着缓存容量增大，幂律拓扑结构的跳数比小于随机拓扑结构。无论缓存容量大小，层次化结构的跳数比一直处于最低水平。因为在层次拓扑结构中，用户到达服务器的最短路径相对固定，沿着网络拓扑层次上溯即可，而在随机拓扑结构和幂律拓扑结构下，最短路径较长，使得跳数比性能较差。

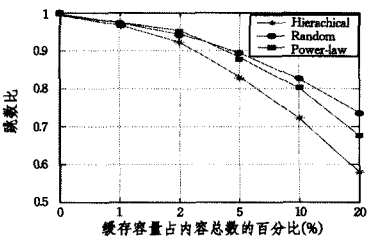

图 12　不同拓扑结构下跳数比的变化曲线

结束语　随着互联网技术与应用的飞速发展，内容相关的流量与日俱增。本文针对信息中心网络的缓存策略建模，指出基于全局内容流行度的替换策略不适用于信息中心网络的分布式模式，进而对信息中心网络中已有的单节点缓存策略和路径缓存算法进行了深入分析，综合“热度”和“新颖度”评价，提出了一种最小活跃度内容替换策略 LAU 和布鲁姆过滤器的数据结构用于统计内容活跃度；并基于该策略，提出了一种自适应路径缓存算法 ACAP。仿真分析表明，ACAP 相比已有的路径缓存算法，无论是单缓存命中率，还是服务器响应命中率和跳数比，都有较优的性能，从而验证了该算法的可用性和合理性。

参 考 文 献

[1] System C. Cisco Visual Networking Index(VNI): Forecast and methodology, 2014-2019[EB/OL]. http://www.cisco.com

[2] Carofiglio G, Gallo M, Muscariello L. Bandwidth and Storage Sharing Performance in Information Centric Networking[C]// Proceedings of ACM SIGCOMM ICN Workshop. 2011: 26-31

[3] Tsilopoulos C, Xylomenos G. Supporting Diverse Traffic Types in Information Centric Networks [C]// Proceedings of ACM

SIGCOMM ICN Workshop. 2011:13-18

[4] Chiocchetti R, Rossi D, Carofiglio G, et al. Exploit the Known or Explore the Unknown? Hamlet-Like Doubts in ICN[C]// Proceedings of ACM SIGCOMM ICN Workshop. 2012:7-12

[5] Jacobson V, Smetters D K, Thronton J D, et al. Networking named content[C]// Proceedings of CoNEXT. 2009:1-12

[6] Chiocchetti R, Perino D, Carofiglio G, et al. INFORM: a dynamic Interest Forwarding Mechanism for Information Centric Networking[C]// ACM SIGCOMM Workshop on Information-centric Networking. HongKong, China, 2013:9-14

[7] Ardelius J, Grönvall B, Westberg L. On the Effects of Caching in Access Aggregation Networks[C]// Proceedings of ACM SIGCOMM ICN WorkShop. 2012:67-72

[8] Zhu Yi, Mi Zheng-Kun, Wang Wen-Nai. A Cache Probability Replacement Policy Based on Content Popularity in Content Centric Networks[J]. Journal of Electronics & Information Technology, 2013, 35(6):1305-1310(in Chinese)
朱轶,糜正琨,王文鼐. 一种基于内容流行度的内容中心网络缓存概率置换策略[J]. 电子与信息学报,2013,35(6):1305-1310

[9] Laoutaris N, Che H, Stavrakakis I. The lcd interconnection of lru caches and its analysis[J]. Perform. Eval., 2006, 63(7):609-634

[10] Psaras I, Chai W K, Pavlou G. Probabilistic In-Network Caching for Information-Centric Networks[C]// Proceedings of ACM SIGCOMM ICN WorkShop. 2012:55-60

[11] Chai W K, He D, Psaras I, et al. Cache 'Less for More' in Information-centric Networks[C]// Proc. of IFIP Networking. 2012:27-40

[12] Fricker C, Robert P, Roberts J, et al. Impact of traffic mix on caching performance in a content-centric network[C]// IEEE INFOCOM, Workshop on NOMEN. 2012:310-315

[13] Pacifici V, Dán G. Content-peering Dynamics of Autonomous Caches in a Content-centric Network[C]// IEEE INFOCOM. Turin, 2013:1079-1087

[14] Liu Wai-xi, Yu Shun-zheng, Cai Jun, et al. Scheme for cooperative caching in ICN[J]. Journal of Software, 2013, 24(8):1947-1962(in Chinese)
刘外喜,余顺争,蔡君,等. ICN 中的一种协作缓存机制[J]. 软件学报,2013,24(8):1947-1962

[15] Liu Wai-xi, Yu Shun-zheng, Hu Xiao, et al. Selective caching in content-centric networking[J]. Chinese Journal of Computers, 2013, 37(2):275-288(in Chinese)
刘外喜,余顺争,胡晓,等. CCN 中选择性缓存机制的研究[J]. 计算机学报,2013,37(2):275-288

[16] Cui Xian-dong, Liu Jiang, Huang Tao, et al. A Novel In-network Caching Scheme Based on Betweenness and Replacement Rate in Content Centric Networking[J]. Journal of Electronics & Information Technology, 2014, 36(1):1-7(in Chinese)
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

[17] Ge Guo-dong, Guo Yun-fei, Lan Ju-long, et al. Collaborative Caching Algorithm Based on Request Correlation in Named Data Networking[J]. Journal of Electronics & Information Technology, 2014, 36(12):2795-2801(in Chinese)
葛国栋,郭云飞,兰巨龙,等. 命名数据网络中基于内容请求相关性的协作缓存算法[J]. 电子与信息学报,2014,36(12):2795-2801

[18] Ge Guo-Dong, Guo Yun-Fei, Liu Cai-Xia, et al. A Collaborative Caching Strategy for Privacy Protection in Content Centric Networking[J]. Journal of Electronics & Information Technology, 2015, 37(5):1220-1226(in Chinese)
葛国栋,郭云飞,刘彩霞,等. 内容中心网络中面向隐私保护的协作缓存策略[J]. 电子与信息学报,2015,37(5):1220-1226

[19] Ge Guo-dong, Guo Yun-fei, Lan Ju-long, et al. Dynamic cache size transfer scheme based on replacement rate in content centric networking[J]. Journal on Communications, 2015, 36(5):120-129(in Chinese)
葛国栋,郭云飞,兰巨龙,等. CCN 中基于替换率的缓存空间动态借调机制[J]. 通信学报,2015,36(5):120-129

[20] Cai Jun, Yu Shun-zheng, Liu Wai-xi. Caching strategy based on node's importance to community in information-centric networks[J]. Journal on Communications, 2015, 36(6):173-182(in Chinese)
蔡君,余顺争,刘外喜. 基于节点社团重要度的 ICN 缓存策略[J]. 通信学报,2015,36(6):173-182

[21] Hu Qian, Wu Mu-qing, Liu Hong-bao. Distributed Cooperative Caching Strategy in Content centric Networking[J]. Journal of Beijing University of Posts and Telecommunications, 2015, 38(2):98-103(in Chinese)
胡骞,武穆清,刘红宝. 内容中心网络中基于分布式协作的缓存策略[J]. 北京邮电大学学报,2015,38(2):98-103

[22] Zhang Guo-qiang, Li Yang, Lin Tao, et al. Survey of In-Network Caching Techniques in Information-Centric Networks[J]. Journal of Software, 2014, 25(1):154-175(in Chinese)
张国强,李杨,林涛,等. 信息中心网络中的内置缓存技术研究[J]. 软件学报,2014,25(1):154-175

[23] Zhang G Q, Li Y, Lin T. Caching in information centric networking: a survey[J]. Computer Networks, 2013, 57(16):3128-3141

[24] Caroglio G, Gallo M, Muscariello L, et al. Modeling data transfer in conten-centric networking [C]// Proceedings of International Teletraffic Congress. 2011:111-118

[25] Chen Kang, Yu Hong-liang, et al. Adaptive replication management algorithm based on location and file popularity in peer-to-peer networks[J]. Chinese Journal of Computers, 2009, 32(10):1927-1937(in Chinese)
陈康,余宏亮,等. 对等网络中基于位置信息和文件流行度的自适应复本管理算法[J]. 计算机学报,2009,32(10):1927-1937

[26] Yu H, Zheng D, Zhao B Y, et al. Understanding user behavior in large scale video on demand systems[C]// Proceedings of the 1st EuroSys Conference(EuroSys'06). Leuven. Belgium, 2006:333-344

[27] Saar C, Yossi M. Spectral Bloom filters[C]// Proc. of the 2003 ACM SIGMOD Int'l Conf. on Management of Data. San Diego: ACM, 2003:241-252

[28] Borst S, Gupta V, Walid A. Distributed caching algorithms for content distribution networks[C]// Proceedings of IEEE INFOCOM. 2010:4244-5837