软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2013,24(8):1947−1962 [doi: 10.3724/SP.J.1001.2013.04378]

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

ICN 中的一种协作缓存机制*

刘外喜[1,2], 余顺争[1], 蔡 君[3], 高 鹰[2]

[1](中山大学 电子通信工程系,广东 广州 510006)

[2](广州大学 电子信息工程系,广东 广州 510006)

[3](广东技术师范学院 电子与信息学院,广东 广州 510665)

通讯作者: 刘外喜, E-mail: liuwaixi@gmail.com, http://www.gzhu.edu.cn

摘 要: 为了克服现有 Internet 架构存在的众所周知的缺点,未来网络的研究成为热点.ICN(information-centric networking)在众多新架构中正逐渐被公认为最有前途的方案.它把传输的内容缓存到沿途的节点.高效的缓存机制是它的一个重要研究方面.为此,提出了一种在分布式缓存机制中嵌入中心式缓存决策的机制(content-aware placement,discovery and replacement,简称 APDR),它把内容的放置、发现、替换统一起来考虑,实现内容的有序缓存,提高网络的性能.APDR 的主要思想是:Interest 报文除了携带对内容的请求以外,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 Interest 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 Data 报文上,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.在多种实验条件下对 APDR 进行了仿真验证,结果表明,APDR 可以改善网络性能,包括缓存命中率、接入代价、替换数量、转发效率以及缓存鲁棒性等;而且 APDR 的额外开销也不大.

关键词: ICN(information-centric networking);内容放置;内容替换;承诺时间;差异化缓存

中图法分类号: TP316 **文献标识码**: A

中文引用格式: 刘外喜,余顺争,蔡君,高鹰.ICN 中的一种协作缓存机制.软件学报,2013,24(8):1947−1962. http://www.jos.org.cn/1000-9825/4378.htm

英文引用格式: Liu WX, Yu SZ, Cai J, Gao Y. Scheme for cooperative caching in ICN. Ruan Jian Xue Bao/Journal of Software, 2013,24(8):1947−1962 (in Chinese). http://www.jos.org.cn/1000-9825/4378.htm

Scheme for Cooperative Caching in ICN

LIU Wai-Xi[1,2], YU Shun-Zheng[1], CAI Jun[3], GAO Ying[2]

[1](Department of Electronics and Communication Engineering, Sun Yat-Sen University, Guangzhou 510006, China)

[2](Department of Electronic and Information Engineering, Guangzhou University, Guangzhou 510006, China)

[3](School of Electronic and Information, Guangdong Polytechnic Normal University, Guangzhou 510665, China)

Corresponding author: LIU Wai-Xi, E-mail: liuwaixi@gmail.com, http://www.gzhu.edu.cn

Abstract: With the emergence of information-centric networking (ICN) in which in-networking caching distinguishes it from other Internet architecture, efficient caching becomes increasingly attractive, but remains of great challenge as current Web caching. This paper proposes a distributed scheme that is embedded with a locally central model, called APDR (i.e., content-aware placement, discovery and replacement). In APDR, according to the information carried on Interest message, the destination of the Interest makes caching decision for the nodes along the path, including different time that the requested content will be cached in different nodes. Finally, the study evaluates the proposed scheme through extensive simulation experiments in terms of a wide range of performance metrics. The experiment results show that the scheme can yield a significant performance improvement over diverse operating environments, such as,

* 基金项目: 国家自然科学基金(U0735002, 60970146, 61202271); 广东省自然科学基金(S2011040004068, S2012040007184)
收稿时间: 2012-08-14; 修改时间: 2012-11-14; 定稿时间: 2013-01-25

cache hit ratio, average access cost, number of replacement, forwarding efficiency and cache robustness etc. At the same time, additional overhead of APDR is small.

Key words: information-centric networking; content placement; content replacement; commitment time; differentiated caching

当前,各种新的互联网应用层出不穷,但现在的 TCP/IP 协议架构却无法适应应用需求的发展,在移动性支持、网络扩展性、安全性等方面存在着一系列的问题.为了彻底地解决这些问题,设计一种全新的互联网架构逐渐成为研究者们的共识.Web 缓存是一项改善 Web 服务的重要技术.它使得内容更加靠近用户,从而可以减少网络带宽的消耗、服务器的负载、用户的访问时延[1].正是考虑到这些优势,在设计全新未来互联网的时候,这一针对 Web 服务的思想被扩展到全网以及所有应用.ICN(information-centric networking)[2]正是这一思想的典型代表,它正逐渐被大家认为是众多未来互联网架构中最有前途的一种方案.

ICN 要求每个节点都缓存经过的内容,覆盖全网的缓存成为网络体系结构中固有的一部分,网络不仅是一个传输体,而且成为一个存储体.当用户请求某一内容时,任何缓存有该内容的中间节点都可以做出响应,而不是像传统节点那样直接转发.这样的全网缓存机制使得信息快速地扩散到网络中,对于每一个请求,网络都可以提供多个数据源,从而大幅度提高网络的性能.这实际上将以前只有 CDN(content distribution network)和 P2P(peer-to-peer)等专有网络中才会提供的多方通信服务扩展到全网络.

若要最大程度地发挥覆盖全网的缓存所带来的好处,设计一个高效的缓存机制就成为关键.高效的缓存意味着:在部署相同大小的缓存的情况下,用户能够更容易地从附近节点的缓存中获取想要的内容,而不需要总是从原始内容服务器那里获取.但当前的 ICN 缓存机制仍然存在以下问题:

- 同质化缓存

对于非协作式缓存,各个节点独立地缓存和替换将会导致内容的分布存在以下不合理:

(1) 各节点缓存相同的内容;

(2) 内容在空间分布上太集中或太分散,这样将会导致请求者需要从过于集中的或分散的节点上获取内容,导致流量不合理;

(3) 在时间上分布不合理,在热门时间里,每个节点都缓存相同的内容,然而热门时间一过,该内容在各节点上又几乎同时消失了.

以上不合理将导致缓存的效率不高,网络缓存的潜能不能完全地发挥出来.

- Interest 扑空

对于非协作式缓存,各节点之间互相不知道对方的缓存内容;而在协作式缓存中,即使各节点互相知道对方缓存的内容,但没有时间承诺,并且各节点的替换是独立的,所以内容随时有可能被替换掉.当中间节点盲目地或者根据没有时间承诺的缓存信息向某一节点转发 Interest 的时候,该目标内容可能已经没有了,Interest 会白跑一趟而只能重新路由到其他节点,并且这一次还是和前一次一样得不到保证.这使得缓存的作用具有一定的随机性和偶然性,Interest 的转发效率变得较低.

为了解决同质化缓存和 Interest 扑空等问题,我们提出了 APDR(content-aware of placement,discovery and replacement).本文的主要贡献如下:

(1) 提出了一种差异化缓存机制,避免了同质化缓存;

(2) 通过采用承诺缓存时间的方法,内容的放置、内容的发现、内容的替换被统一起来考虑,即实现了有序的缓存,避免了 Interest 扑空;

(3) 定义一系列适合 ICN 这一新架构的性能测量参数,并通过广泛的实验验证了 APDR 的优越性能.

1 相关工作

自 Web 缓存技术提出以来,已经涌现出了许多机制,研究领域包括内容的放置(placement)和替换(replacement)策略等.在放置方面,包括处处缓存[2]、选择性缓存[3]等方法;在替换方面,实现方法主要是利用用户

对内容的请求所具有的时间局域性(temporal locality)和空间局域性(spatial locality)来尽可能地提高替换的准确性,从而提高缓存的效率,如 LRU(latest recently used),LFU(least frequently used)等[1].如果抛开研究对象、体系架构以及具体技术的差异,这些机制可分为两类:协作式和非协作式.其中,协作式又可分为显式协作(explicit cooperation)和隐式协作(implicit cooperation).下面综述一些与本文相关的协作式机制.

1.1 显式协作(explicit cooperation)

显式协作的一个显著特点就是:节点之间互相通告自己所缓存的内容等信息.

Yang 等人提出了 PPP(pull with piggybacked push)协议[4],其基本思想是 Push 和 Pull 结合.Push- based,当自己有内容时,就向邻居 push 广告信息,告诉别人自己有哪些内容;Pull-based,当一个节点需要某个内容时,就先在自己的 zone 内广播 request 包,如果没有回应再直接找服务器.为了节省通信开销,将 push 的广告信息附带到 request 包中.PPP 采用处处缓存的思想;内容替换策略如下:zone 中缓存该内容的节点数量越多,越优先被替换.PPP 提高了节点之间缓存的共享效率,但是 PPP 依然没有根本性地消除无效 push 和额外延迟增加等问题.并且,PPP 是针对 MANET 这一特定应用场景提出来的,是否适用于其他场景还需进一步考证.

在使用优化的思想实现将内容放置在更加合适的节点的问题上,Shen 等人[5]在线性拓扑上进行了研究:当 client 向服务器端请求内容 O 时,将沿途各节点的信息(目标内容的大小($s(O)$)、访问频率($f(O)$)、因为替换而需要付出的代价($m(O)$)等)带到服务器端.服务器端根据这些信息计算得到一个最合适缓存的节点的集合,目标是使全网用户对所有目标内容的获取代价(access cost)最小.这个计算结果会由逆向沿途返回的 Data 报文带到各个节点,各节点据此更新自己的缓存.该方法属于选择性缓存.在替换策略上,使用贪心算法来选择替换效率最高的目标.在此基础上,Li 等人[6]讨论了树型拓扑网络中的内容放置的优化问题.显而易见的是,因为采用了集中式的优化计算,这类方法能够很好地得到合适的放置地点,但 $s(O)$,$f(O)$,$m(O)$的精确度极大地影响着该方法的效果.而实际上这些信息并不容易获取,要精确就需要额外开销(如文献[7]利用内容 O 在过去被请求的历史来进行推测);而要节省开销则会以降低性能为代价.同时,这类方法的计算量也不小,被用到的动态规划算法的复杂度达到 $O(k^2)$,k 是沿途存有描述符的节点数量,接近于沿途节点的数量.

文献[8]采用了处处缓存;节点在做替换决定的时候,会和邻居交换 CCP(cached content propagation)来了解对方缓存的内容,优先替换掉邻居依然存有的内容.这样做的好处是可以实现内容的差异化缓存,而缺点是每一次替换所引发的 CCP 交换势必会带来较大的通信开销,对于缓存有限、内容多的网络更加受限.Wang 等人[9]也提出了在节点之间交换缓存信息的思想:(1) 每个节点承诺缓存时间;(2) 利用 bloom filter 技术,内容目录在一定区域内被广播给其他人,被广播的区域大小是根据缓存的承诺时间,其承诺时间越长,被广播的距离越远.但在文献[9]中没有给出如何来确定承诺时间的方法,并且也没有通过实验验证其思想.Ming 等人[10]也提出了内容年龄的概念,即流行度高以及放置于网络边缘的内容的年龄更长,目的是使流行度高的内容更加靠近网络边缘,从而减少用户的延迟和网络流量.但该文假设网络拓扑和内容的流行度都已知,而实际上这两种信息的获知并不容易.

Bekta 等人[11]针对已知的 CDN 网络拓扑以及用户需求模式进行内容放置和路由的联合优化,目标是使用户总的接入成本最小.文献[11]首次将内容放置与路由两个子问题之间互相耦合,使得目标函数是一个二次整数非线性规划,并提出了两种有效的线性化算法.但该文运算规模较大(其实验设计也证明了这一点),只能作为一个静态规划的解决方案,无法实时地应对用户需求随时-空变化的特点.

李春洪等人[12]针对多媒体内容分发服务中节点负载不平衡的问题,设计了一种无热点的覆盖网协同缓存策略——HFOCC.通过将“热点”对象复制到低负载节点,分散服务请求,达到消除热点的目的.文献[12]采用一种“软”副本生命期控制机制,当工作负载发生变化时,冗余副本被及时删除.该文从平衡负载的角度考虑缓存的优化放置问题,牺牲少量的命中率换取吞吐率和资源利用率.节点需要记录每个内容在网络中所有的位置以便于查询,在内容比较多时,查询速度以及建立该记录所需要的通信开销都是需要考虑的问题.

1.2 隐式协作(implicit cooperation)

隐式协作是在避免通信开销的基础上实现协作缓存.

Chai 等人[3]提出的基于介数(betweenness)的选择性缓存机制:只将内容缓存于 Interest 沿途中介数最大的节点,即,使内容缓存于更重要的节点上,从而实现更高的击中率,并减少替换次数.这种选择性缓存机制存在的问题是:介数最大节点的缓存空间被挤满了,而其他节点的缓存空间却没有被充分利用.文献[3]的替换策略是LRU.Fiore 等人[13]提出了 Hamlet 算法:利用无线信道的广播特性,如果节点 n 偷听到节点 m 在请求内容 r,n 就认为 m 自此以后会缓存内容 r;如果节点 n 偷听到节点 p 回应了内容 r 的请求,n 就认为 p 拥有内容 r.在知道了周围节点缓存什么内容后,自己在做替换决定的时候就优先保存周围节点没有的内容[13].文献[13]虽然实现了差异化缓存,但可以看出,偷听信息是这种方法的实现基础,所以不可靠,具有随机的波动性,无法避免甚至会加剧 Interest 扑空现象.

Rosensweig 等人提出了 Breadcrumbs 算法[14]:通过记住目标内容的五元组,每个节点记住该内容被转发与缓存的历史.五元组包括:内容的上游、下游节点号、内容经过该节点的最近时间、内容在该节点被请求的最近时间等.利用这些信息就可以分析出内容在网络中的缓存情况,从而可以将内容的放置、发现统一起来.但Breadcrumbs 有可能会出现循环路由,并且记住每个内容历史的开销也不小.Hosseini-Khayat 等人[15]从合适地选择缓存对象的角度将内容的放置和替换一起统筹考虑.其中心思想是:如果缓存一个新到的内容需要替换掉一个对提升系统性能更有“价值”的内容,就没有必要缓存.但该方法中的“价值”依赖于对流行度指标 T 的判断,而文中的 T 是一个预设的固定值,所以不能成为一种在线的解决方案.

总的来说,显式协作比隐式协作具有更好的性能,但必须为此而付出一定的通信或计算开销.

此外,吴建平等人[16]对未来互联网进行了综述,提出了互联网体系结构可演进性评估模型的若干思路.同时,针对 ICN 框架下的缓存理论研究,Rosensweig 等人[17]提出了近似地模拟多级缓存网络的 *a-NET* 模型;Carofiglio 等人[18]研究了在 ICN 中同时有多个应用时,缓存策略对性能的影响;Psaras 等人[19]研究了 ICN 中的缓存动力行为.

APDR 和以前的工作不同之处在于:通过采用承诺缓存时间的方法,实现差异化缓存;内容的放置、内容的发现、内容的替换被统一起来考虑,并且都是有序的.APDR 也摆脱了前述相关工作所存在的一些约束:这些工作需要预先知道网络拓扑、内容流行度等先验知识.APDR 能够根据网络拓扑以及内容潜在需求动态地调整自己的策略;同时,APDR 也是一种通用的不针对任何特殊应用场景的机制.

2 ICN 概览

在过去几年里,有多个国家设立研究项目展开对 ICN 的研究,提出了不少框架体系,主要有 NDN[20],CCN[2],PSIRP(现在是 PURSUIT)[21],DONA[22]等等.本文的机制是在 CCN 框架的基础上提出来的.在本节,我们将综述 ICN 的一些重要内容,更加详细的内容参见文献[2].

所有 ICN 架构都有两个共同特点:通信是由接收者(内容需求者)发起的;缓存是全网所有节点的必备功能.ICN 有两类报文:兴趣(interest)和数据(data)报文.为了获得一个内容,需求者发出一个带有内容名字的 Interest.该 Interest 不断地在网络中被转发,直到有中间节点的缓存或内容服务器对其响应.然后,Data 报文会沿着 Interest 报文走过的路径逆向回到需求者.在本文中,我们称这一路径为 Interest 路径.对内容 O 的请求记为 *Interest*(O),来自众多需求者的 *Interest*(O)路径实际上构成了一棵多播树.由于 ICN 中的节点并没有地址,所以整个过程的路由都是基于内容名字的.同时,为了加强网络的安全,在整个转发过程中,每个数据报文总是和内容产生者的签名信息绑定在一起的.这样,中间节点和内容需求者都可以对其进行验证[23].

在ICN中,为了实现内容的路由和转发,每个路由器都有3个功能模块:FIB(forwarding information base),PIT(pending interest table),CS(content store)[2].其中,FIB 和 TCP/IP 架构中路由器的转发表的功能基本类似,但它允许匹配多个端口进行转发.

3 APDR 的结构

假定网络是由原始内容服务器 OCS(original content server)和一些配有缓存的路由器节点以及一些用户组成,其中,只有 OCS 会产生内容,并且会永久地保存该内容.

3.1 APDR的运行机制

与 ICN 一样,当一个节点收到请求同一内容 O 的多个 *Interest*(O)时,只有第 1 个到达的 *Interest*(O)会被转发到上游节点,其他后到的 *Interest*(O)的转发都会终止于该节点.该节点用 PIT 记录需要内容 O 的需求端口.当上游节点响应第 1 个 *Interest*(O)并返回了内容 O 时,该节点将内容 O 多播给这些需求端口.显然,当 PIT 记录表明有多个需求端口时,该节点是 *Interest*(O)构成的多播树上的一个分叉点.由于在 APDR 中,树根和分叉节点都执行相同的 APDR 算法,为了方便起见,我们把该多播树上的根和分叉点统称为根/分叉点.

设 u_i 表示节点 i 的端口总数;$m_i(O)$表示内容 O 在节点 i 已被请求过的端口数量;我们用 $k_i(O)=u_i-m_i(O)$表示节点 i 的下游节点在未来一段时间内对内容 O 的潜在需求数量;用 f_i 表示节点 i 的空闲缓存大小.这些参数都是节点 i 在向上转发 *Interest*(O)时携带的实时信息.

如图 1 所示,在 APDR 中,当内容需求者节点 S 请求内容 O_1 时,*Interest*(O_1)路径上的 $k_i(O_1)$和 f_i 等信息将会被 *Interest*(O_1)携带到根节点 D.节点 D 将根据这些信息计算出缓存策略摘要 CSS(caching strategy summary)[24]. CSS 指明 *Interest*(O_1)路径上哪些节点必须缓存 O_1 以及承诺缓存多久.该 CSS 会被 Data 报文带回到沿途各个节点,各个节点根据 CSS 管理自己的缓存.同时,各个节点根据 CSS 也能知道沿途哪些节点缓存内容 O_1 以及缓存多久,以帮助后到的 Interest 在附近找到内容 O_1,因而可以实现有序的内容发现.

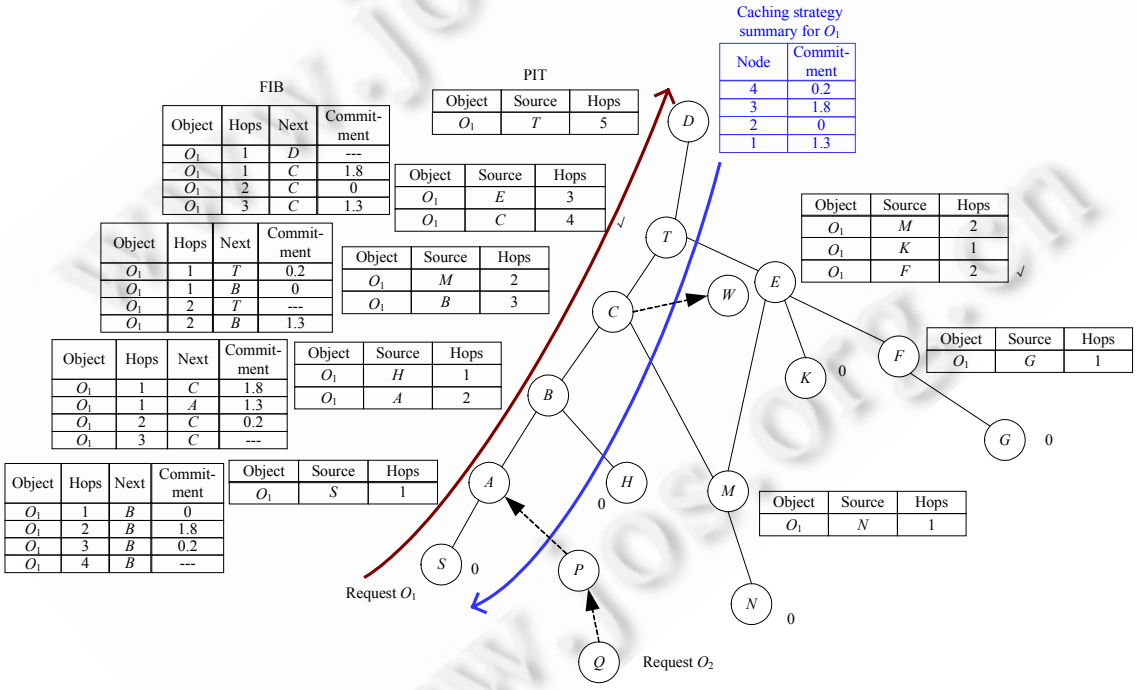


Fig.1 Communication scheme of APDR

图 1 APDR 的通信机制

APDR 的运行机制还包括以下几个方面:

1) 缓存空间管理的原则:定期替换+LFU.在 APDR 中,缓存承诺时间到期的内容会被标记为“可替换”;如有多个“可替换”的内容,最少被访问的将优先被替换.设节点 i 的缓存容量为 C_i,已被占用的空间为 s_i,

其他正在请求的内容预期需要的空间为 r_i,“可替换”的缓存空间 e_i,则空余的缓存空间 $f_i=C_i-s_i+e_i-r_i$.

2) 制定 CSS 的原则:量身定做.当携带 CSS 的 Data 报文到达途中每个节点时,如果其 PIT 中记录着多个端口对该内容有需求,即该节点是 Interest 树上一个分叉点,则该 CSS 只代表了通过其中一个端口的一条路径的缓存策略;而通过其他端口的其他路径的 CSS,该节点则重新计算,以保证 CSS 的适应性.例如图 1 中的节点 *T*,它在收到来自 *D* 的 CSS(代表 *D-T-C-B-A-S* 路径上的 CSS)之后,会在节点 *T* 重新针对 *T-E-F-G* 计算 CSS,并把它附加在向 *E* 转发的 Data 报文上.

3) Interest 转发原则:最长路径的优先.当同时有多个下游方向的 Interest 到达节点时,只选择来自最长路径方向 Interest 向上转发.例如图 1 中的节点 *T*,将会把来自 *S-A-B-C-T* 的 Interest 优先转发到上游节点.如果多个方向具有相同的路径长度,那么随机地选择其中一个,例如图 1 中的节点 *E*.在选择 Interest 之后,都需要在 PIT 中做好标记,以方便 CSS 的正确返回,如图 1 中的标记“√”.在向上转发 Interest 之后,对于后来到达的 Interest,即使路径更长,也不需要上报.

由此可见,在 APDR 中,中间节点根据 CSS 去确定是否缓存以及缓存的时间.通过给路径上不同的节点分配不同的缓存时间,实现差异化缓存.同时,因为提出了承诺时间的概念,每个节点都可以预计目的地的内容是否过期,那么 Interest 就不会出现“Interest 空”了.并且,每个节点只替换过期的内容.所以,这样也就实现了有序的内容放置、发现和替换.

3.2 APDR功能模块的定义

如图 1 所示,为了实现 APDR,需要对 ICN 中的功能模块稍作修改,本节给出它们的功能定义.

- CSS 的定义

CSS 是 APDR 新提出来的功能单元,包括以下字段:

- Node:对 Interest 路径上各节点的相对位置进行标识的 ID,内容需求者是其计数的基准节点.
- Commitment:各节点对内容进行缓存需要做出的时间承诺.

- PIT 的定义

在每一个节点上,PIT 的作用是记住那些 Interest 在该节点没有被响应而需要被转发到其他节点的来源端口.这样,当数据报文到达时,就可以从其 Interest 的来源端口返回,以保证数据报文沿着正确的 Interest 路径逆向返回.

主要包括以下字段:

- Object:Interest 所需目标内容的名字.
- Source:Interest 的来源端口.
- Hops:本节点距离内容需求者的跳数,该字段主要是帮助节点知道自己在整个 Interest 路径的相对位置.它与 CSS 中的 Node 字段配合,就可从 CSS 中知道自己是否应该缓存以及缓存多少时间.

- FIB 的定义

FIB 是由基于内容名字的路由协议收敛后得到的,在本文中,我们假设路由协议已经收敛得到了基本的 FIB.APDR 为 FIB 增加了 Hops 和 Commitment 两个字段,利用 CSS 对其进行更新,使其更加准确,主要包括以下字段:

- Object:Interest 所需目标内容的名字.
- Hops:目标内容距离本节点的跳数.
- Next:去往目标内容的路由的下一跳.
- Commitment:目标内容所在节点对其缓存的承诺时间.它来源于 CSS 的 Commitment.

3.3 CSS算法

CSS 是实现整个 APDR 的关键,本节介绍与 CSS 相关的主要内容.

3.3.1 制定 CSS 的基本思想

APDR 中制定 CSS 的基本思想是基于以下考虑:

对于内容 O,节点 i 上的某个端口如果曾经收到过对该内容的请求,并且已经把内容 O 从这个端口转发出去,那么因为下游节点缓存的存在,在未来一段时间内就不会再次从这个端口收到对该内容的请求.所以,当节点 i 的所有端口都请求过内容 O 并被满足以后,节点 i 则没有必要缓存内容 O.因此,节点 i 上没有请求过内容 O 的端口数量越少,在未来一段时间内,节点 i 收到对内容 O 的请求的概率就会越低,节点 i 缓存内容 O 的承诺时间也就可以越短.

OCS 会永久地保存内容 O,所以距离 OCS 较近的节点缓存内容 O 的承诺时间可以短一些,因为即使 *Interest*(O)在该节点没有被响应,*Interest*(O)被转到 OCS 的代价也不高.

如果将流行度高的内容尽可能地缓存在网络边缘节点,即,它们缓存这些内容的承诺时间长一些,则可以更好地服务用户,缓存的效率会提高.但要实时、准确地确定内容的流行度则比较困难,我们采用以下渐进式的方法来确定:如果内容 O 是真正流行的,则内容 O 会被来自不同位置的用户多次请求 *Interest*(O).在此过程中,那些缓存着内容 O 的节点既是 *Interest*(O)的终点(即 *Interest*(O)路径的终点),也是制定 CSS 的节点.所以,如果每个终点每次制定 CSS 时让 *Interest*(O)路径上的中间节点承诺缓存最长的时间,那么,*Interest*(O)路径逐步地向用户方向迁移将会是一个大概率事件;那么,被选中用来缓存内容 O 最长时间的节点也就会逐步地向网络边缘迁移.这样,一种多次的、渐进式的过程可以将真正流行的内容筛选出来,并被长时间地缓存于网络边缘.

3.3.2 根/分叉点制定 CSS 的算法

下面的算法 1 给出了制定 CSS 的伪代码,一共由 4 大步组成:

- 第 1 步,确定 $T_i(O)$,目的是要让 *Interest*(O)路径上的有空闲缓存空间的节点对内容 O 的承诺缓存时间成山峰形状.即,中间节点承诺缓存时间最长,越往两端承诺时间越短.
- 第 2 步,确定 CSS 中的潜在需求因子.我们把各节点的潜在需求占总需求的百分比作为潜在需求因子,其中,$P(i,O)$表示内容 O 在节点 i 的潜在需求因子.
- 第 3 步,确定 CSS 中的距离因子,越靠近 OCS,距离因子越小.其中,$D(i,O)$表示内容 O 在节点 i 的距离因子.
- 第 4 步,确定各节点最终的缓存时间,其中,$C_i(O)$表示内容 O 在节点 i 的缓存时间;β和γ分别是潜在需求因子和距离因子的调节参数.通常令$\beta>\gamma$,即优先满足潜在需求.β和γ的最佳值可以通过实验确定,例如,$\beta=1,\gamma=0.8$.

算法 1. 每一个根/分叉点制定 CSS 的算法.

第 1 步. 确定 $T_i(O)$.

从 Interest 携带的信息中获得 $k_j(O)$和 $f_j,j=1,...,N$,其中,N 是 Interest 路径长度,j 是节点距离需求者的距离;由 $\{f_j\}$和内容 O 的大小确定 Interest 路径上哪些节点可以缓存该内容,设有 n 个节点可以缓存内容 O;用 i 表示节点在这 n 个节点中的顺序,j_i 是其与需求者的距离,i 越小离需求者越近.

For $i=1:n$

  IF $i\leqslant n/2$ **then** $T_i(O)=i$; //逐步地增加承诺时间

  IF $i>n/2$ **then** $T_i(O)=n+1-i$; //逐步地减少承诺时间

End For

第 2 步. 确定 CSS 中的潜在需求因子.

$$P(i,0)=\frac{k_i(O)}{\sum_{i=1}^{n}k_i(O)}(i=1,2,...,n).$$

第 3 步. 确定 CSS 中的距离因子.

$$D(i,0)=-\frac{1}{d+N}(d+j_i)(i=1,2,...,n).$$

其中,d 是当前制定 CSS 的节点与 OCS 之间的距离,由 FIB 可知.

第 4 步. 确定各节点最终的缓存时间.

For i=1:n

$C_i(O)=[1+\beta\times P(i,O)+\gamma\times D(i,O)]\times T_i(O)$

End For

下面是一个制定 CSS 过程的例子.为简单起见,这里的缓存大小和缓存承诺时间使用的都是单位大小和单位时间.如图 1 所示,节点 S 请求内容 O_1,设内容 O_1 的大小为 5,$Interest(O_1)$到达目的地 D 时,也就建立了一个 $Interest(O_1)$路径(A-B-C-T),即 N=4.节点 D 制定了一个如图 1 所示的 CSS.计算过程见表 1,由于节点 B 的空闲缓存 f_2 小于 5,所以不能缓存内容 O_1,即,参与 CSS 计算的只有 3(n=3)个节点(A,C,T).节点 A 和 C 分别有一个对内容 O_1 的潜在需求端口($k_1(O_1)$=1;$k_3(O_1)$=1).从计算结果可知,A 节点承诺缓存时间为 1.3;C 节点承诺缓存时间为 1.8;T 节点承诺缓存时间为 0.2.包括 B 节点在内,$Interest(O_1)$路径上的每个节点都根据该 CSS 更新自己的缓存以及 FIB.

Table 1 An example of making CSS

表 1 一个制定 CSS 的例子

节点	j	i	$k_i(O_1)$	f_i	$P(i,O_1)$	$D(i,O_1)$	$T_i(O_1)$	$C_i(O_1)$
A	1	1	1	10	1/2	−1/4	1	1.3
B	2		0	4	0	−2/4	0	0
C	3	2	1	6	1/2	−3/4	2	1.8
T	4	3	0	10	0	−4/4	1	0.2

注释:β=1;γ=0.8;n=3;N=4;d=0.

4 仿真实验

如第 1 节所述,与本文相关的研究包括内容的放置和替换,我们从中选择代表性的策略作为 APDR 性能比较的对象:一个是传统的处处缓存的放置策略与 LFU 的替换策略的组合,称为 CEE+LFU;另一个是最近提出来的一个选择性缓存机制——基于介数的放置策略[3],它与 LRU 的替换策略的组合称为 Betw+LRU.本文采用多个性能参数将 APDR 与 CEE+LFU 以及 Betw+LRU 进行对比,其中包括缓存命中率、平均接入代价、替换数量、内容差异率等.另外,我们也研究了各个网络参数对性能的影响,如缓存大小(cache size)、内容数量(number of content)、节点数量、Zipf 参数(α)等.

4.1 性能参数

ICN 根本性地改变了网络架构以及通信模式,所以需要定义一些能够更加准确地反映 ICN 性能的测量参数,本文对此进行了尝试.

- 缓存命中率(cache hit ratio)

缓存命中率是一个典型的测量缓存性能的参数.它被定义为由缓存而不是 OCS 响应用户请求的概率.缓存命中率越高,缓存系统的效率就越高.

- 平均接入代价(average access cost)

它被定义为 Interest 找到所需求内容的平均距离,本文以跳数(hop)为单位.

- 替换数量(number of replacement)

一般情况下,互联网中的缓存大小显然会远远小于内容的大小,这就势必会需要大量的替换操作,为此也就会消耗大量的计算资源.而对于一个资源有限的路由器来说,这些消耗显然将会影响路由器的转发等其他任务的完成.从另一个方面来说,过多的替换也会消耗大量的能源,这也违背了绿色网络的精神.所以,如何减少 ICN 中的替换数量,也是一个需要值得注意的问题.

APDR 与其他缓存策略的差异之一在于每个节点对待一个新到达内容的处理方法.对于 CEE+LFU,每个新到达的内容都会被缓存起来,也就会引起一次替换;反之,在 APDR 中,每个节点根据 CSS 来确定是否缓存该内

容,每个节点只替换掉那些过期的内容.显然,这种策略将会减少替换数量;而 Betw+LRU 则每次只在沿途中介数最大的节点缓存,所以它的替换数量会进一步地减少.为了定量地测量上述分析,我们定义了替换数量:一个 Interest 在一个节点上引起的替换次数,它显然是一个归一化的参数.

- 内容差异率(content diversity ratio,简称 CDR)

实现差异化缓存从而提高全网的整体性能,是 APDR 的主要目标.而在 CEE+LFU 中,由于各节点的替换决定是独立做出的,所以,同质化缓存非常严重.为了定量地比较这种差异程度,我们定义内容差异率如下:

$$CDR=\frac{t_1}{t_2}.$$

其中,t_1 表示缓存中的所有内容的种类数量,t_2 表示 OCS 产生的所有的内容种类的总数量.

CDR 实际上测量的是缓存中所存内容的重复程度.显然,CDR 值越大,差异程度越大.

- Interest 命中率(interest hit ratio)

在 ICN 中,可以采用两种转发策略:基于 FIB 和泛洪.前者显然可以有效地减少 Interest 的转发数量,但也许会错过一些更近的响应位置;后者可以减少 FIB 的规模以及不断更新 FIB 而需要的开销,但其无法恒定地保证最优的分发性能[25].无论是哪种策略,我们都希望 Interest 的转发效率高一些.也就说,我们希望每一次转发都尽最大可能地帮助 Interest 靠近所需的内容,而不是 Interest 扑空.对于泛洪机制来说,因为网络需要大量地转发 Interest,高效的 Interest 转发效率显得更加重要.为了测量 Interest 的转发效率,我们定义 Interest 命中率如下:

$$IHR=\frac{n_1}{n_2}.$$

其中,n_1 是 Interest 被缓存响应的数量,n_2 是发送的 Interest 的数量.

4.2 实验参数设置

我们的实验网络拓扑[26]由 100 个节点和 386 条链路组成.它由广域网和城域网组成,广域网是骨干网,城域网完成接入的任务.内容的请求过程服从泊松过程,亦即请求的间隔时间是指数分布,平均是 1~10 个单位时间.同时,我们假设用户的访问模式遵循 Zipf 分布[27],就是说,如果用 $\Pr\{C_k\}$表示第 k 级受欢迎程度的内容被请求到的概率,那么它遵循以下规律:$\Pr\{C_k\}\propto k^{-\alpha}$,$\alpha$被称为 Zipf 参数(Zipf parameter (α)).表 2 列出了本文主要的实验参数以及默认值.

Table 2 Parameters of experiments

表 2 实验参数

参数	默认值	变动范围
WAN 节点数/MAN 节点数	4:96	
用户的数量	50 000	
内容的数量	2 000	100~5 000
内容的大小(MB)	平均为 1	1~5
节点缓存的大小(MB)	10	1~80
访问模式	Zipf: α=0.7	0.1~1

4.3 实验结果

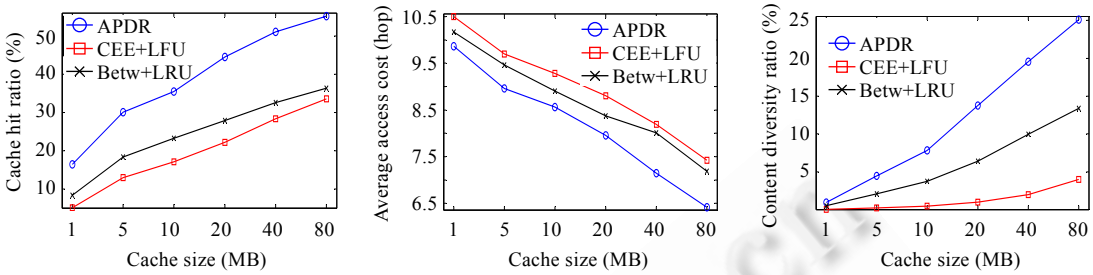
本节给出实验结果,如果没有特别说明,本节所有结果都是我们经过 10 次实验后得到的平均.同时,为了观察网络性能受某个参数的影响,我们一次只让 1 个参数变化,其他参数保持不变,其取值如表 2 默认值所示.

4.3.1 缓存大小的影响

本节首先研究缓存大小对系统性能的影响.图 2 显示出系统性能随缓存大小变化而变化的情况,其中,缓存大小从 1MB 一直增加到 80MB.从图 2(a)可以看到,正如所预料的那样,3 种机制的缓存命中率会随着缓存的增加而增加,但在这个过程中,APDR 的性能一直好于 CEE+LFU 和 Betw+LRU.

图 2(b)显示出,3 种机制的平均接入代价会随着缓存的增加而减小.这也很容易解释,因为缓存命中率提高

了,接入代价自然也就下降了.APDR 的平均接入代价自始至终都比 CEE+LFU 和 Betw+LRU 要低.从图 2(c)可知,3 种机制的 CDR 会随着缓存的增大而增大,但 APDR 总是比 CEE+LFU 和 Betw+LRU 要大.也正是由于 APDR 的差异化缓存的实现,使得 APDR 的 CDR 增速比 CEE+LFU 和 Betw+LRU 的快.



(a) 缓存命中率 vs. 缓存大小 (b) 平均接入代价 vs. 缓存大小 (c) 内容差异率 vs. 缓存大小

Fig.2 Effect of cache size

图 2 缓存大小的影响

表 3 呈现的替换数量随缓存大小变化的结果与第 4.1 节的理论分析相吻合:Betw+LRU 在 3 种机制中的替换数量最小;CEE+LFU 中每一个 Interest 都会引起一次替换;即使 APDR 的替换数量随缓存变大而增加,但其依然远远小于 1.APDR 的替换数量随着缓存数量的增加而逐步增加,这主要是因为 APDR 的工作机制使然:缓存空间越大,参与 CSS 制定的节点就越多;随后,内容过期的节点数就会越多,替换数量也就越多.也正是经过更多的替换,才使得那些用户真正需求的内容被保留在缓存中,从而成就了更高的缓存命中率.

同时应该注意的是,APDR 相对于 CEE+LFU 在替换数量上的节省是在改善其他缓存性能的基础上获得的.表 4~表 6 也展现了相似的结论,限于篇幅,后文不再赘述.

Table 3 Number of replacement vs. cache size

表 3 替换数量 vs. 缓存大小

Cache size (MB)	APDR	CEE+LFU	Betw+LRU
1	0.000 59	1	0.003 35
5	0.005 03	1	0.003 23
10	0.011 59	1	0.003 16
20	0.031 60	1	0.003 04
40	0.070 52	1	0.002 89
80	0.140 47	1	0.002 68

Table 4 Number of replacement vs. number of content

表 4 替换数量 vs. 内容数量

Number of content	APDR	CEE+LFU	Betw+LRU
100	0.004 28	1	0.001 10
200	0.004 44	1	0.001 15
500	0.004 74	1	0.001 53
1 000	0.007 97	1	0.002 18
2 000	0.013 02	1	0.003 16
5 000	0.023 59	1	0.004 56

Table 5 Number of replacement vs. number of nodes

表 5 替换数量 vs. 节点数量

Number of nodes	APDR	CEE+LFU	Betw+LRU
5	0.015 86	1	0.004 35
10	0.014 86	1	0.003 81
20	0.011 55	1	0.003 21
40	0.009 05	1	0.002 65
60	0.008 17	1	0.002 39
100	0.006 82	1	0.002 08

Table 6 Number of replacement vs. Zipf parameter (α)

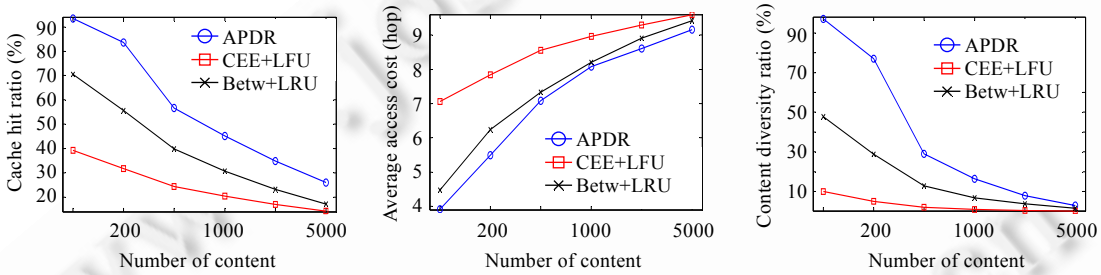
表 6 替换数量 vs. Zipf 参数(α)

Zipf参数(α)	APDR	CEE+LFU	Betw+LRU
0.10	0.017 57	1	0.003 66
0.20	0.016 81	1	0.003 63
0.30	0.015 82	1	0.003 59
0.40	0.014 87	1	0.003 51
0.55	0.013 29	1	0.003 36
0.65	0.013 41	1	0.003 23
0.75	0.012 07	1	0.003 10
0.85	0.011 58	1	0.002 88
0.90	0.010 45	1	0.002 80
1.00	0.009 90	1	0.002 61

4.3.2 内容数量的影响

图 3 给出了系统性能随内容数量变化而变化的情况,内容数量大范围地从 100 变化到 5 000 个.

由图 3(a)可知,3 种机制的缓存命中率随着内容数量的增加而显著地减少.这主要是因为缓存的大小没变而内容的数量增加了,缓存就相对变得越发稀缺,Interest 被缓存响应的概率自然也就会降低.但无论如何,APDR 与 CEE+LFU 和 Betw+LRU 相比始终显现出明显的优势.在图 3(b)中,正如所预料的那样,3 种机制都会随着内容数量的增加而需要更大的平均接入代价,这也是伴随缓存命中率下降的必然结果,但 APDR 始终比 CEE+LFU 和 Betw+LRU 要低.图 3(c)显示,3 种机制的 CDR 会随着内容数量的增加而减少,而 APDR 在各种情况下都始终好于 CEE+LFU 和 Betw+LRU.不过,APDR 的相对优势随着内容数量的增加而在逐步缩小.



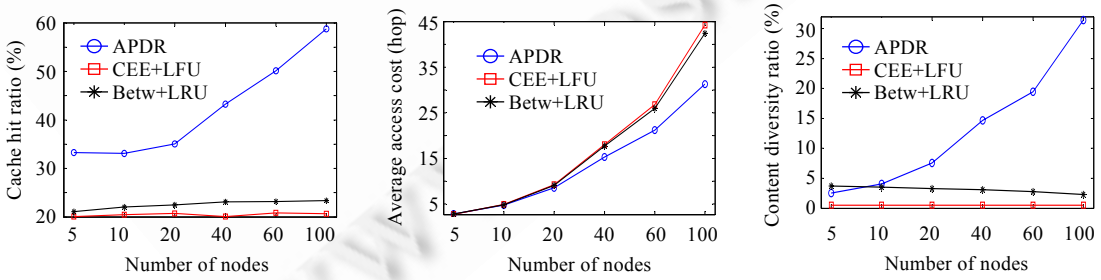
(a) 缓存命中率 vs. 缓存大小 (b) 平均接入代价 vs. 缓存大小 (c) 内容差异率 vs. 缓存大小

Fig.3 Effect of number of content

图 3 内容数量的影响

4.3.3 节点数量的影响

为了研究网络的规模对缓存机制的影响,我们从网络中选取多个不同长度的链路作为性能测量对象.图 4 是实验结果,其中,横轴是被选取链路的节点数量.



(a) 缓存命中率 vs. 缓存大小 (b) 平均接入代价 vs. 缓存大小 (c) 内容差异率 vs. 缓存大小

Fig.4 Effect of number of nodes

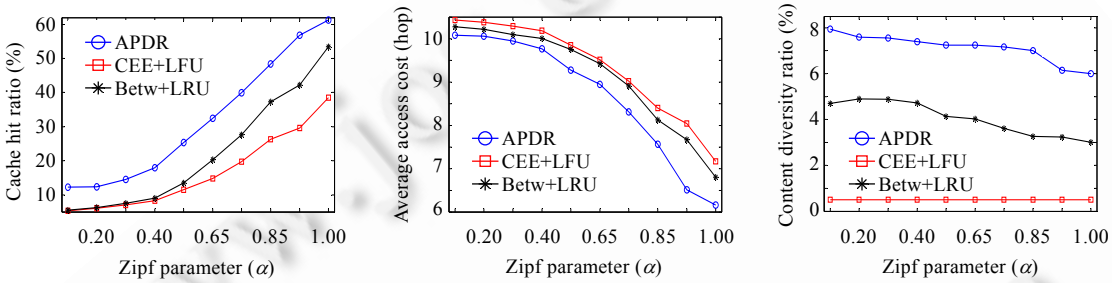
图 4 节点数量的影响

对于 APDR 来说,由于差异化缓存的实现,更多的节点意味着更多的缓存空间,也就意味着更多种类的内容被缓存.但对于 CEE+LFU 来说,每个节点都独立地决定自己的缓存,同质化缓存严重,所以即使有再多的节点提供再多的缓存也无法提高内容差异率.图 4(c)很好地证明了这一点,随着节点数量的增加,CEE+LFU 和 Betw+LRU 的 CDR 几乎保持不变,而 APDR 的 CDR 在增加.

这也就很好地解释了图 4(a)所看到的结果,因为随着 APDR 的 CDR 的提高,更多种类的内容缓存于网络中,缓存命中率自然就会提高.这也进一步证明了 APDR 的差异化缓存实现了节点之间缓存的互通有无、互利互惠,从而改善了性能.同时,图 4(a)中也显示,CEE+LFU 和 Betw+LRU 的缓存命中率随着节点数量的增加而几乎保持不变,这是与其不变的 CDR 一脉相承的.图 4(b)则揭示出以下事实:对于 APDR,随着节点数量的增加,缓存内容的种类增加了,但地点也更加分散了,所以虽然 Interest 可以更高概率地从缓存中得到响应,但其需要更远的距离;而对于 CEE+LFU 和 Betw+LRU,伴随着不变的命中率以及内容缓存地点的分散化,其接入代价增加得更加明显.

4.3.4 Zipf 参数(α)的影响

现在,大家已经普遍认为,用户对内容的偏好(即访问模式)是服从 Zipf 分布的[27].不同应用的分布参数也不一样,Zipf 参数(α)越大,说明用户的偏好越集中.在本节,我们研究用户偏好对缓存机制性能的影响,更准确地说,我们想了解缓存机制面向不同应用的表现.图 5 是实验结果,其中,Zipf 参数(α)从 0.1 变化到 1.



(a) 缓存命中率 vs. 缓存大小 (b) 平均接入代价 vs. 缓存大小 (c) 内容差异率 vs. 缓存大小

Fig.5 Effect of Zipf parameter (α)

图 5 Zipf 参数(α)的影响

图 5(a)明显地显示出,3 种机制的缓存命中率会随着 Zipf 参数的增加而增加,即性能会随着内容的时间局域性的加强而改善.这是因为 3 种机制在放置或替换策略上都利用了内容的时间局域性,这也进一步证明了利用内容的局域性在设计缓存机制中的重要性.从图 5(b)可以看到,3 种机制的平均接入代价都会随着 Zipf 参数的增加而降低,但 APDR 一直保持着优势.从图 5(c)可知,随着 Zipf 参数的增加,APDR 和 Betw+LRU 的 CDR 会缓慢地减少,而 CEE+LFU 的 CDR 几乎保持不变,但 APDR 一直高于 CEE+LFU 和 Betw+LRU.

以上的实验结果表明:在各种实验条件下,APDR 的性能都要明显地好于 CEE+LFU,包括缓存命中率、平均接入代价、内容差异率等指标,也更加节能,如替换数量等指标.除了替换数量,APDR 的其他指标也好于 Betw+LRU.

4.3.5 差异化缓存的效果

正如从上文的实验结果很明显地看到的那样,APDR 的缓存命中率和 CDR 之间存在着很强的正相关,其相关系数达到了 0.892 5.图 6 也很好地佐证了这一点.

我们再看看 APDR 的 CDR 与其他参数之间的关系:替换数量与 CDR 的相关系数是–0.25,平均接入代价和 CDR 的相关系数是–0.51.所以说,CDR 与其他参数之间呈现了强相关性.这也再一次证明差异化缓存是提高性能的主要原因.这实际上也是一种潜在的改善性能的方法:设法提高缓存差异率.

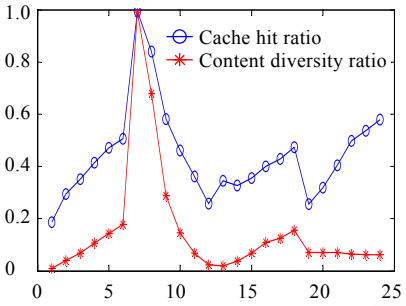


Fig.6 Correlation between cache hit ratio and content diversity ratio

图 6 缓存命中率与 CDR 的相关性

4.3.6 Interest 转发效率的提高

在基于泛洪路由协议下,本节对 APDR,CEE+LFU,Betw+LRU 的 Interest 转发效率做出比较.图 7 是 Interest 命中率随 Zipf 参数从 0.1 变化到 1 的情况.

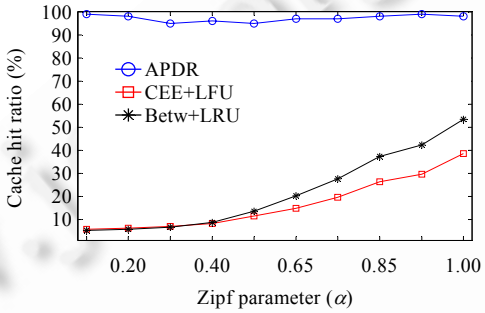


Fig.7 Interest hit ratio vs. Zipf parameter (α)

图 7 Interest 命中率 vs. Zipf 参数(α)

APDR 的 IHR 接近于 100%,波动幅度很小.这主要是因为 Interest 从 FIB 知道哪些目的地的内容还没有过期,它可以从中有选择性地泛洪.而 CEE+LFU 是盲目地泛洪,会出现 Interest 扑空现象,很多 Interest 被转发出去后根本就不会被响应;Betw+LRU 也存在类似的问题,所以它们的 IHR 小于 100%.APDR 自始至终都好于 CEE+LFU 和 Betw+LRU,APDR 大幅度地提高了 Interest 的转发效率.Interest 命中率随其他参数的变化也有类似结论,限于篇幅,这里不再赘述.实际上,这也是所有那些没有时间承诺的缓存机制的通病.

4.3.7 缓存鲁棒性的改善[28]

在 ICN 中,内容可以被缓存在路由器中,于是缓存的鲁棒性问题也就伴随而来.

缓存污染攻击(cache pollution attack)[29]是威胁缓存鲁棒性的主要来源,其主要思想是:破坏内容的局域性,使得缓存中保留着一些虚假的受欢迎的内容,从而降低缓存的效率.研究表明:在缓存污染攻击中,敌手的请求不遵循正常的 Zipf 分布而是均匀分布的随机访问模式,这样会导致缓存系统的性能急剧下降[28].

为了评估 APDR 和 CEE+LFU 在性能上的差距,本文定义如下参数:

$$Gap=\frac{S_{APDR}}{S_{CEE+LFU}},$$

其中,S_{APDR} 表示 APDR 的性能,$S_{CEE+LFU}$ 表示 CEE+LFU 的性能.

图 8 是各种性能 Gap 随 Zipf 参数变化的情况,从中我们可以明显地看到,APDR 在各种性能参数上都比 CEE+LFU 具有明显的优势.最值得注意的是,当 Zipf 参数很小时,即内容的局域性被破坏了,APDR 的优势依然巨大,这也从侧面反映出 APDR 防御缓存污染攻击的能力较强,改善了缓存鲁棒性.APDR 与 Betw+LRU 的比较

结果也类似,在此不再赘述.

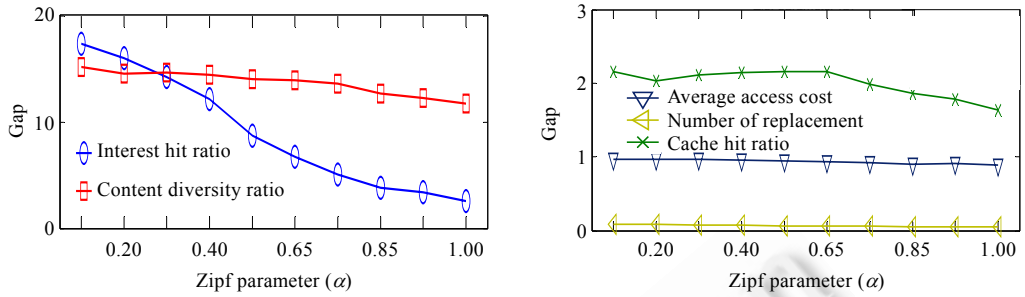


Fig.8 Gap vs. Zipf parameter (α)

图 8 Gap vs. Zipf 参数(α)

4.4 APDR的开销

上文的实验结果表明 APDR 具有良好的性能表现,而为此付出的开销代价也不大,包括通信和计算开销:

- 通信开销:随 Interest 上传的 $k_i(O)$和 f_i,及随 Data 下传的 CSS 是其主要开销.在本文中,$k_i(O)$需要 4bit,f_i需要 8bit,CSS 仅需要 16bit,这些 bit 级别的通信开销显然都不大.
- 计算开销:分析算法 1 可知,其时间复杂度仅为 $O(n)$,其中,n 是 Interest 路径上有空闲缓存的节点的数量,是一个较小的值.据我们分析,在一个具有 100 个节点的网络中,n 一般都不超过 8,所以 APDR 的计算开销也很小.

表 7 的数据是在 2.8GHz 的 CPU(Intel Pentium 4)和 2G 内存的 PC 机上测量得到的,所以 APDR 完全可以满足高速网络中线速转发的要求.

Table 7 Computing overhead of APDR

表 7 APDR 的计算开销

Interest路径的节点数量	计算时间(s)	计算速度(Mbps)
5	0.000 030	400
10	0.000 039	307
20	0.000 048	250
50	0.000 058	206

也就是说,APDR 以较小的开销换取了良好的性能.

4.5 潜在需求的确定

在本文中,$k_i(O)$表示 Interest 在节点 i 被转发给上游节点时的潜在需求数量,是一个实时数据.由于节点 i 在收到第 1 个 *Interest*(*O*)之后会查询 FIB,PIT,CS 等以确定是否向上游转发该请求,在此期间,可能会从其他端口陆续收到多个 *Interest*(*O*).所以,$k_i(O)$实际上反映的是这段时间的瞬时潜在需求.由于没有特意等待更长一段时间,所以 $k_i(O)$并不能准确地反映未来一段时间内节点 i 的实际潜在需求.因此,根/分叉点据此制定的承诺时间只能反映节点 i 未来潜在需求的上限.从另一个角度说,如果节点 i 等待一段时间再转发 *Interest*(*O*),虽然可以提高 $k_i(O)$的准确性,但在总体上就会造成 *Interest*(*O*)的较大延迟,对性能的负面影响会更大.节点 i 也可以根据历史数据预测其未来的潜在需求,以提高 $k_i(O)$的准确性,但会带来算法的复杂性,增加节点的运算负担.实验结果也表明,采用 $k_i(O)$的瞬时值,对性能的改善更为明显.

5 结 论

为了使 ICN 的全网缓存发挥出潜在的优势,本文提出了一种在分布式缓存方案中嵌入中心式协作缓存机制的 APDR,使得内容的放置、内容的发现、内容的替换被统一起来考虑,实现了有序的内容缓存策略.实验结

果表明,APDR 与其他方法相比具有以下优势:

(1) 更好的缓存性能,包括缓存命中率、平均接入代价、内容差异率等;

(2) 更高的 Interest 命中率,即更高的 Interest 转发效率;

(3) 改善了缓存鲁棒性;

(4) APDR 在多种实验条件下都有良好的性能,具有良好的扩展性.

进一步的研究将主要从以下两个方面展开:

- 首先,把本文的缓存机制放到实际的环境(如 PlanetLab)下进行实验验证,并实现算法的优化;
- 其次,将 APDR 拓展到移动网络环境中,并研究如何将推送机制嵌入到 APDR 之中.

References:

[1] Wang J. A survey of Web caching schemes for the Internet. ACM SIGCOMM Computer Communication Review, 1999,29(5): 36−46. [doi: 10.1145/505696.505701]

[2] Jacobson V, Smetters DK, Thornton JD, Plass MF, Briggs NH, Braynard RL. Networking named content. In: Proc. of the CoNEXT 2009. ACM Press, 2009. 1−12. [doi: 10.1145/1658939.1658941]

[3] Chai WK, He DL, Psaras I, Pavlou G. Cache “less for more” in information-centric networks. In: Bestak R, *et al*., eds. Proc. of the NETWORKING 2012. LNCS 7289, 2012. 27−40.

[4] Yang KT, Chiu GM. A hybrid pull-based with piggybacked push protocol for cache sharing. The Computer Journal, 2011,54(12): 2017−2032. [doi: 10.1093/comjnl/bxr079]

[5] Shen H, Xu SH. Coordinated en-route Web caching in multiserver networks. IEEE Trans. on Computers, 2009,58(5):605−619. [doi: 10.1109/TC.2008.162]

[6] Li KQ, Shen H, Chin FYL, Zheng SQ. Optimal methods for coordinated enroute Web caching for tree networks. ACM Trans. on Internet Technology (TOIT), 2005,5(3):480−507. [doi: 10.1145/1084772.1084774]

[7] Tang X, Chanson ST. Coordinated en-route Web caching. IEEE Trans. on Computers, 2002,51(6):595−607. [doi: 10.1109/TC.2002.1009146]

[8] Raychaudhuri LDYZ. Enhance content broadcast efficiency in routers with integrated caching. In: Proc. of the IEEE ISCC 2011. IEEE, 2011. 320−322. [doi: 10.1109/ISCC.2011.5983797]

[9] Wang YG, Lee KH, Venkataraman B, Shamanna RL, Rhee I, Yang S. Advertising cached contents in the control plane: Necessity and feasibility. In: Proc. of the IEEE INFOCOM 2012 Workshop on Emerging Design Choices in Name-Oriented Networking. Orlando: IEEE, 2012. 286−291. [doi: 10.1109/INFCOMW.2012.6193507]

[10] Ming Z, Xu M, Wang D. Age-Based cooperative caching in information-centric networks. In: Proc. of the IEEE INFOCOM 2012 Workshop on Emerging Design Choices in Name-Oriented Networking. IEEE, 2012. 268−273. [doi: 10.1109/INFCOMW.2012.6193504]

[11] Bektaş T, Cordeau JF, Erkut E, Laporte G. Exact algorithms for the joint object placement and request routing problem in content distribution networks. Computers & Operations Research, 2008,35(12):3860−3884. [doi: 10.1016/j.cor.2007.02.005]

[12] Li CH, Feng GF, Gu TC, Lu SL, Chen DX. A hotspots-free overlay cooperative caching scheme. Ruan Jian Xue Bao/Journal of Software, 2008,19(3):744−754 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/19/744.htm [doi: 10.3724/SP.J.1001.2008.00744]

[13] Fiore M, Casetti C, Chiasserini CF. Caching strategies based on information density estimation in wireless ad hoc networks. IEEE Trans. on Vehicular Technology, 2011,60(5):2194−2208. [doi: 10.1109/TVT.2011.2136363]

[14] Rosensweig EJ, Kurose J. Breadcrumbs: Efficient, best-effort content location in cache networks. In: Proc. of the IEEE INFOCOM. IEEE, 2009. 2631−2635. [doi: 10.1109/INFCOM.2009.5062201]

[15] Hosseini-Khayat S. Improving object cache performance through selective placement. In: Proc. of the 24th IASTED Int’l Conf. on Parallel and Distributed Computing and Networks (PDCN 2006). 2006.

[16] Wu JP, Lin S, Xu K, Liu Y, Zhu M. Advances in evolvable new generation Internet architecture. Chinese Journal of Computers, 2012,35(6):1094−1108 (in Chinese with English abstract).

[17] Rosensweig EJ, Kurose J, Towsley D. Approximate models for general cache networks. In: Proc. of the IEEE INFOCOM. IEEE, 2010. 1−9. [doi: 10.1109/INFCOM.2010.5461936]

[18] Carofiglio G, Gehlen V, Perino D. Experimental evaluation of memory management in content-centric networking. In: Proc. of the IEEE ICC. IEEE, 2011. 1−6. [doi: 10.1109/icc.2011.5962739]

[19] Psaras I, Clegg RC, Landa R, Chai WK, Pavlou G. Modelling and evaluation of CCN-caching trees. In: Domingo-Pascual J, *et al*., eds. Proc. of the NETWORKING 2011. LNCS 6640, 2011. 78−91. [doi: 10.1007/978-3-642-20757-0_7]

[20] http://www.named-data.net/

[21] http://www.psirp.org/

[22] Koponen T, Chawla M, Chun BG, Ermolinskiy A, Kim KH, Shenker S, Stoica I. A data-oriented (and beyond) network architecture. In: Proc. of the SIGCOMM 2007. ACM Press, 2007. 181−192. [doi: 10.1145/1282380.1282402]

[23] Ghodsi A, Koponen T, Raghavan B, Shenker S, Singla A, Wilcox J. Information-Centric networking: Seeing the forest for the trees. In: Proc. of the 10th ACM Workshop on Hot Topics in Networks (HotNets-X). ACM Press, 2011. 1−6. [doi: 10.1145/2070562.2070563]

[24] 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]

[25] Chiocchetti R, Rossi D, Rossini G, Carofiglio G, Perino D. Exploit the known or explore the unknown? Hamlet-like doubts in ICN. In: Proc. of the ACM SIGCOMM 2012 Workshop on Information-Centric Networking. 2012. 7−12. [doi: 10.1145/2342488.2342491]

[26] Calvert KI, Doar MB, Zegura EW. Modeling Internet topology. IEEE Communications Magazine, IEEE, 1997,35(6):160−163. [doi: 10.1109/35.587723]

[27] Lee B, Pei C, Li F, Graham P, Scott S. Web caching and Zipf-like distributions: Evidence and implications. In: Proc. of the IEEE INFOCOM. IEEE, 1999. 126−134. [doi: 10.1109/INFCOM.1999.749260]

[28] Xie MJ, Widjaja I, Wang HN. Enhancing cache robustness for content-centric networking. In: Proc. of the IEEE INFOCOM. Orlando: IEEE, 2012. 2426−2434. [doi: 10.1109/INFCOM.2012.6195632]

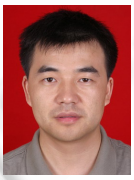
[29] Gao Y, Deng LW, Kuzmanovic A, Chen Y. Internet cache pollution attacks and countermeasures. In: Proc. of the IEEE ICNP. IEEE, 2006. 54−64. [doi: 10.1109/ICNP.2006.320198]

附中文参考文献:

[12] 李春洪,冯国富,顾铁成,陆桑璐,陈道蓄.一种无“热点”的覆盖网协同缓存策略.软件学报,2008,19(3):744−754. http://www.jos.org.cn/1000-9825/19/744.htm [doi: 10.3724/SP.J.1001.2008.00744]

[16] 吴建平,林嵩,徐恪,刘莹,朱敏.可演进的新一代互联网体系结构研究进展.计算机学报,2012,35(6):1094−1108.

刘外喜(1976－),男,湖南茶陵人,博士生,讲师,CCF 学生会员,主要研究领域为未来网络,网络安全,网络编码.
E-mail: liuwaixi@gmail.com



蔡君(1981－),男,博士,讲师,主要研究领域为复杂网络,网络安全.
E-mail: gzhcaijun@gmail.com

余顺争(1958－),男,博士,教授,博士生导师,CCF 会员,主要研究领域为未来网络,网络安全.
E-mail: syu@mail.sysu.edu.cn

高鹰(1963－),男,博士,教授,主要研究领域为智能优化算法,盲信号分离,自适应信号处理,图像识别.
E-mail: falcongao@sina.com.cn