DOI:10.7544/issn1000-1239.2016.20148097
53(6):1281-1291,2016

命名数据网络中一种基于节点分类的数据存储策略

黄 胜 滕明埝 吴 震 许江华 季瑞军
(重庆邮电大学光纤通信技术重点实验室 重庆 400065)
(tmnfighting@163.com)

A Data Caching Scheme Based on Node Classification in Named Data Networking

Huang Sheng, Teng Mingnian, Wu Zhen, Xu Jianghua, and Ji Ruijun
(*Key Laboratory of Optical Fiber Communication Technology, Chongqing University of Posts and Telecommunications, Chongqing* 400065)

Abstract Compared with the traditional Internet, in-networking caching is one of the most distinguishable features in named data networking (NDN). In NDN, a node caches every passing data packet as a default model. The caching scheme generates a large number of redundant data in in-networking. As a consequence, the networking cache resource is wasted seriously. To solve the problem, a caching scheme based on node classification (BNC) is proposed firstly in this paper. Based on different node positions, the nodes that data packet passes through are divided into two types: "edge" type and "core" type. When data packet passes through the "core" type nodes, by considering location and data popularity distribution at different nodes, it is cached in a node which is beneficial to other nodes. When the data packet passes through the "edge" nodes, a node is selected through data popularity to be beneficial to the client. The simulation results show that the proposed scheme can efficiently improve the in-network hit ratio and reduce the delay and hops of getting the data packet.

Key words named data networking (NDN); node classification caching scheme; in-network caching; redundant data; content centric networking

摘 要 缓存是命名数据网络(named data networking, NDN)有别于传统网络最突出的特性之一,NDN中默认所有节点都具有缓存所有经过数据的功能.这种"处处缓存"策略导致网内大量冗余数据的产生,使网内缓存被严重浪费.针对上述问题,首次提出了一种基于节点分类(based on node classification, BNC)的数据存储策略.基于节点位置的不同,将数据返回客户端所经过的节点分为"边缘"类节点与"核心"类节点.当数据经过"核心"类节点时,通过权衡该类节点的位置与数据在不同节点的流行度分布,将数据存储在对其他节点最有利的节点中;当数据经过"边缘"类节点时,通过该数据流行度来选择最有利于客户端的位置.仿真结果表明,提出的策略将有效提高数据命中率,减少数据请求时延和距离.

关键词 命名数据网络;节点分类数据存储策略;网内存储;冗余数据;内容中心网络

中图法分类号 TP393

收稿日期:2014-10-13;**修回日期**:2015-03-26

基金项目:国家自然科学基金项目(61371096,61275077);国家"九七三"重点基础研究发展计划基金项目(2012CB315803);重庆市自然科学基金项目(cstc2013jcyjA40052);重庆市教委科学技术研究项目(KJ130515)
This work was supported by the National Natural Science Foundation of China (61371096,61275077), the National Basic Research Program of China (973 Program) (2012CB315803), the Natural Science Foundation of Chongqing (cstc2013jcyjA40052), and the Scientific and Technological Research Program of Chongqing Municipal Education Commission (KJ130515).

通信作者:滕明埝(tengmingnianzbb@163.com)

随着互联网的发展,传统的 TCP/IP 网络暴露出越来越多的缺陷,主要表现为可扩展性较差、安全可控性低、移动性支持不足等,从而,严重制约了互联网的发展[1].因此,引发了人们对未来互联网架构的研究热潮.许多研究机构先后提出了不同的新型网络架构,诸如斯坦福大学的 TRIAD[2]和加州大学伯克利分校的 DONA[3],及其他未来互联网架构(XIA[4],4WARD[5],SAIL[6],COMET[7]).2009 年 Jacobson 等人[8]提出内容中心网络(content centric networking, CCN),其中,命名数据网络(named data networking, NDN)作为内容中心网络中最具代表性的一种新型网络架构得到了广泛的研究.在文献[9]中,分别对 NDN 的缓存、路由以及部署等方面进行了详细分析.当缓存一定时怎样分配缓存到不同的节点上,使整个网络的性能最优等问题在文献[10]中也有详细的分析.除以上研究外,以数据为中心的应用也逐渐得到了广泛的研究[11].本文在节点缓存已经分配好的情况下对数据缓存在那些节点上使整个网络性能得到提升进行研究.

与传统基于 host-to-host 的网络模型相比,NDN 将内容作为主体,不再关心内容存储的位置,而关心的是内容本身是什么[12-16].NDN 中每个节点都具有内容存储库(content storage, CS),这是其有别于传统网络最大的特点.与传统网络相比客户端所请求的数据来源不仅仅是原始内容服务器(original content servers, OCS),也可以是网内任意缓存有对应数据的网内节点.因此,将数据缓存在什么位置使整个网络的性能最佳已经成为了制约 NDN 发展的关键因素之一.

在早些时候 Laoutaris 等人提出了 LCD(leave copy down)[17-18]与 MCD(move copy down)[17-18]以及 Prob(copy with probability)[17]三种策略,其中,LCD 策略将数据仅在命中节点的下一节点缓存;MCD 策略除将数据从命中节点移向其下一跳节点以外,还将删除数据在命中节点中的副本;Prob 策略使每个节点都以概率 p 缓存经过的数据,其中,$0<p<1$.这 3 种策略在一定程度上都减少了网内缓存中的冗余数据,LCD 策略与 MCD 策略也隐含地考虑了数据的请求流频率,当数据请求频率较高(数据比较流行)时会使数据靠近客户端缓存.但是,3 种策略将数据经过的所有节点都"一视同仁",因此,会导致对高性能节点的利用不充分.其中,LCD 策略与 MCD 策略往往会使靠近服务器的节点负载远远高于远离服务器的节点,从而造成节点负载失衡.

刘外喜等人[19]提出 SC 策略,该策略认为数据传播应分为传播早期与晚期 2 个阶段,其中,早期客户端对数据的需求高于晚期,因此,在这 2 个时期应该使用不同的缓存机制.在数据传播早期由于网内节点对数据的缓存较少,此时 SC 策略对数据块在网络中的合理分布有一定的帮助.但是,SC 策略忽略了用户请求数据时具有实时性与随机性的事实,因此,将用户对数据的请求分为 2 个时期的方式不能很好地体现用户请求数据的特性.

Chai 等人[20]提出了 Betw 策略,该策略通过计算节点的介数来判断节点的重要性(介数越高的节点越重要).当数据返回客户端的过程中,只将数据缓存在介数最大的节点上.这个机制简单易行,在一定程度上使整个网络性能得到了提升.但是,Betw 策略将使介数较大的节点十分"拥挤",这些节点的负载将远远高于其他低介数节点,并且,高介数节点中的数据替换频繁.这样即使具有较高流行度的数据也很容易被替换,使下一次该数据的请求不能被满足.

Li 与 Wu 等人[21]从降低 ISP 间流量和用户平均访问时延的角度出发,提出了基于节点分层与分级的优化策略.该策略依据数据流行度的不同,将数据缓存在不同的层、级上,从而使整个网络的性能最佳,但是本文的拓扑分层与分级思想在除树形拓扑以外的拓扑中实现较为困难,因此,其扩展性有待提高.

Psaras 与 Chai 等人[22]提出了 ProbCache 策略,该策略依据请求所经过链路的节点的缓存能力与节点到客户端节点的距离,得到每个节点缓存数据的概率值,然后以此概率值作为节点缓存数据的依据.该策略在一定程度上减小了网内数据的冗余度,提高了缓存的利用效率,使整个网络的性能有所提高.但是,ProbCache 策略没有考虑数据的流行度因素,将所有的数据都等同看待,从而使流行的数据没有得到更好的利用.

针对上述所存在的问题,本文首次将数据包返回客户端所经过的节点分为不同类型,对不同类型的节点按照不同的存储策略对数据进行存储,并考虑了用户请求数据时的实时性,本文所提出的基于节点分类(based on node classification, BNC)机制非常简单,不需要复杂的运算,而且数据包与兴趣包所携带的附加字段大小比较固定,不会因为数据的大小变化而变化,完全符合缓存策略"从简"的要求.

1　NDN 缓存策略常见问题

为了充分利用节点的缓存,设计一种高效的数据缓存策略便成为了关键.怎样缓存数据(数据缓存在什么地方)才能使用户请求数据的时延更短、跳数更少、网内命中率更高等便成为了一个高效缓存策略的重要表现.而在当前的缓存设计中主要存在以下需要考虑的问题:

1) 近客户端缓存与中心缓存.为了减少客户端请求数据的时延与跳数,尽可能地将数据缓存在离客户端近的节点上,如果数据的流行度较高,即短时间内数据的请求量很大,这样的缓存方式在一定程度上会减少数据请求的时延.但由于近客户端节点的缓存往往所服务的客户端节点很少,甚至有可能只服务于一个客户端节点,将导致缓存利用率低等问题.反之,中心缓存便是将数据缓存在网络中重要的位置使其为更多的客户端服务,显然,如果缓存在中心位置的数据只针对某一个客户端有高流行度,那么,与近客户端缓存相比将产生更大的请求时延等问题.

2) 无效缓存.NDN 默认采用处处缓存(leave copy everywhere, LCE),即节点将不加区分地缓存所有经过的数据.如图 1 所示,节点 p 为 OCS 节点,当客户端第 1 次请求数据 i 时请求将到达节点 p,如果采用 LCE 数据缓存策略,那么,在数据 i 返回客户端的所有经过节点中都要缓存数据 i.然而,下次请求数据 i 时在节点 c 便可得到相应数据,这样其他节点缓存的数据 i 便成为了"无效数据",从而浪费缓存空间.

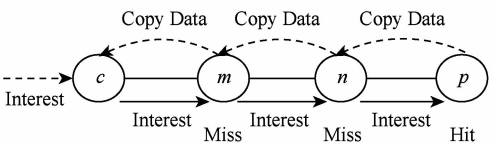


Fig. 1　LCE data packet cache scheme.

图 1　LCE 数据缓存策略

3) 非实时性.由于客户端对数据的请求具有实时性,即在不同时间客户感兴趣的数据不同.但是,大多数缓存策略并没有将其考虑其中,从而导致过时的数据长期占用有限的缓存空间.

针对以上问题,本文提出了 BNC 策略.该策略将数据返回客户端途径的节点分为"边缘"类节点与"核心"类节点(本文后续部分将"边缘"类节点与"核心"类节点分别用Ⅰ类、Ⅱ类表示).当数据经过Ⅰ类节点时,由于该类节点主要服务对象为与其相近的客户端,因此数据根据该客户端的请求流行度对数据进行缓存.当数据经过Ⅱ类节点时,为了使缓存数据的影响范围更广,数据的缓存位置将由节点位置与数据在节点中的流行度分布共同决定,同时为了满足实时性,本文将 2 次数据请求时间间隔作为 BNC 的重要影响因子之一.

2　BNC 数据存储策略运行机制

在本文提出的系统模型中,我们将整个网络标记为 $G(V,E)$,其中 $V=\{v_1,v_2,\cdots,v_n\}$ 表示网络中所有节点的集合,$E=\{e_1,e_2,\cdots,e_m\}$ 表示网络中所有链路的集合.整个网络由内容原始节点与具有 NDN 特性的路由节点及客户端节点组成.

2.1　BNC 基本思想

NDN 中现有的数据缓存策略往往只考虑了某一类节点的缓存情况,如文献[13]中所提出的 Betw 缓存策略只考虑到链路上最大介数节点的缓存,这样将其他次要节点完全否决的思想,将降低整个网络中缓存的利用率.文献[14]也提出了类似节点分类的 ProbCache 缓存策略,但文中的节点分类方式不具有一般性.结合上面文章的不足与启示,本文提出了一种更为合理的 BNC 缓存策略.

NDN 中数据包返回客户端时往往会经过多个中间路由器节点,当一个网络确定时,其节点的位置也就确定了.然而,节点所在的位置不同对整个网络起到的作用也不一样.通常情况下节点越靠近客户端其作用范围越小,越靠近中心的节点其作用范围越广.靠近客户端的节点由于离客户端较近,如果能将客户端请求频率较高的数据存储在这些节点中,会大大减少客户端请求数据的时延与距离.但由于靠近客户端的节点作用的范围有限,从而会导致其他节点不能访问到这些节点中的数据,导致请求的网内命中率下降.然而,中心节点往往能够影响较多的客户端节点,因此,将数据缓存在中心节点处将有效提高请求的网内命中率.

通过上述分析,本文提出的 BNC 策略将数据包返回客户端所经过的节点分为Ⅰ类节点与Ⅱ类节点 2 类,其中Ⅰ类节点为"边缘"节点,Ⅱ类节点为"核心"节点.Ⅰ类节点主要为对应客户端提供高效的数据传输(迅速地满足客户端请求)功能;Ⅱ类节点则是为了更广地服务其他客户端的请求.可以看出

2类的节点作用大不相同,因此,对这2类节点应当采用不同的缓存策略.其中,Ⅰ类节点主要依据客户端节点对数据请求的流行度分布来缓存数据;Ⅱ类节点则主要依据节点位置以及数据在不同节点的流行度不同来缓存数据.从而在满足当前请求数据的客户端的同时也满足其他客户端潜在的需求,并且,为了避免网内产生不必要的冗余数据,每次数据请求过程中,数据在每个类型的节点中只被缓存一次.结果表明,BNC策略将减少用户请求数据的时延与距离,并提高请求的网内命中率.

2.2 BNC策略实现

由于BNC策略将数据包返回时经过的节点分为2种不同的类型,并且2个类型的节点有不同的数据缓存方式.因此,本节将分为2个小节分别对Ⅰ类节点、Ⅱ类节点缓存方式进行描述.

2.2.1 "边缘"类节点(Ⅰ类)存储策略实现

设数据返回客户端所经过的路由器节点总数为n,本文将靠近客户端的l跳节点设置为Ⅰ类节点,而其余$n-l$个节点属于Ⅱ类节点.文中的l参数值主要由客户端到服务器端的平均距离决定,本文中的仿真用到了2种拓扑类型,第2种拓扑结构相对于第1种而言更为复杂,但是由于第2种拓扑增加了客户端节点与服务器端节点的个数,这样一来便使客户端节点到服务器端节点的平均距离十分接近,并且通过实验证明,在本实验中当$l=3$时仿真性能最好(请求平均命中率最高),因此,本文中的$l=3$.如图2所示,$V_1=\{v_1,v_2,v_3\}$表示Ⅰ类节点,$V_2=\{v_4,v_5,\cdots,v_n\}$表示Ⅱ类节点.在本节我们只讨论Ⅰ类节点对数据缓存的实现.

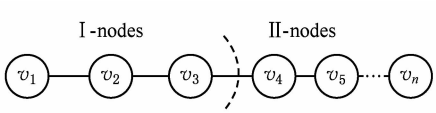


Fig. 2 The node classification.

图2 节点分类

对Ⅰ类节点而言,越靠近客户端越重要.因此,当数据由Ⅱ类节点返回时,该数据将被缓存在节点v_3.因为,已有3个Ⅰ类节点,如果数据请求流行度较高,那么,应该被缓存于Ⅰ类节点中,而不会到Ⅰ类以外的节点请求数据.因此,当数据从Ⅱ类节点返回时将其看作是当前非流行数据.虽说将此类数据定义为非流行数据,但由于其是最近被请求的数据,并不知道在接下来的时间是否流行,所以不能将其抛弃.由上述分析,此时,将该类非流行数据缓存在离客户端最远的Ⅰ类节点(即节点v_3)中,而该数据经过其他Ⅰ类节点时将不再存储.

当请求被Ⅰ类节点(v_1,v_2,v_3)满足,将对应数据返回客户端时需另作处理.假设请求被节点v_3(或v_1、或v_2)满足,说明该数据相对流行.但也需注意到一个问题,如果数据位于命中节点CS缓存队列的前面,意味着此数据即将被替换(本文使用LRU数据替换策略,该策略替换数据时由最前面的数据开始),那么,将此数据定义为次非流行数据.此时该数据不会被返回客户端时经过的其他节点缓存,但由于该数据刚被访问,它将被放在节点v_3的CS队列末尾(设队列大小为m).本文将Ⅰ类节点CS队列中后$0.3m$个数据定义为流行数据,前$0.7m$个数据定义为次非流行数据.如果当请求被Ⅰ类节点满足的同时该数据又是其CS缓存队列后$0.3m$个数据之一,这充分说明该数据在短时间内被连续请求,则该数据在此时为流行数据.因此,将该数据在其命中节点的下一跳节点缓存(即如果请求被v_3满足,且该数据是节点v_3中CS队列后$0.3m$个数据之一,那么,该数据在返回客户端时被缓存在节点v_2上,v_1并不缓存.当然,节点v_2与此相同.如果请求在节点v_1中被满足,那么,只能将数据缓存在节点v_1的CS队列的末尾).在这种情况下,数据被命中节点的下一跳节点缓存以后是否删除命中节点中的副本便成为了需要仔细考虑的地方.此时,当数据命中节点的接入接口大于等于3时该数据将不被删除,因为,这时如果将数据删除,其他客户端节点相应请求到达该节点请求数据时将不会命中,这样使网内命中率下降的同时也会增加其他客户端请求数据的时延.但如果数据命中节点的接入接口小于等于2,那么,数据将被在命中节点删除,因为,此时命中节点中的相应数据将使冗余数据浪费缓存空间.

2.2.2 "核心"类节点(Ⅱ类)存储策略实现

与Ⅰ类节点的存储策略相比,Ⅱ类节点的存储策略相对而言更加复杂.为了更好地理解Ⅱ类节点的存储策略,下面首先给出节点亲密度与介数的定义.

定义1. 节点亲密度.节点亲密度被定义为该节点到达其他节点距离之和的倒数.因此,节点v_i的亲密度为

$$C(i)=\frac{n-1}{\sum_{j=1}^{n}e_{ij}},\tag{1}$$

其中,e_{ij}表示节点v_i与节点v_j之间的最短路径长度.

节点亲密度可以很好地表达节点到达网络中其他节点的难易程度,能够很好地反映本节点对其他节点的作用能力.

定义 2. 介数. 节点介数定义为网络中所有节点对最短路径经过本节点的路径数占所有最短路径数的比例,则节点 v_i 的介数为

$$B(i)=\frac{2\sum\limits_{\substack{s\neq i\neq t\\ s,i,t\in V}}\frac{\sigma_{st}(i)}{\sigma_{st}}}{n(n-1)},\tag{2}$$

其中,σ_{st} 表示顶点 v_s 到顶点 v_t 所有最短路径数;$\sigma_{st}(i)$表示 σ_{st} 中经过节点 v_i 的最短路径数;$n(n-1)/2$ 用来对无向图中节点介数归一化处理,即此时 $B(i)\in[0,1]$.

Ⅱ类节点相对于Ⅰ类节点而言更靠近网络中心位置,因此,其距离数据请求客户端而言较远,如果将其与Ⅰ类节点"一视同仁",那么,将浪费这些节点的位置优势. 由于这些节点位于网络的中心位置,其将作用于更多的客户端节点,因此,数据经过Ⅱ类节点时应该被存储在最重要节点上. 但所有Ⅱ类节点的重要性并不是一样的,本文采用文献[23]中所提供的节点重要性度量公式,对节点重要性进行计算,其表达式为

$$H(i)=B(i)+\sum_{j=1}^{n}\frac{1}{C(j)}B(j),\quad i\neq j,\tag{3}$$

其中,$B(i)$为节点 v_i 的介数,$C(j)$表示节点 v_j 的亲密度,节点 v_j 将自身的紧密度 $1/C(j)$贡献给其相邻节点,而非相邻的节点则贡献度为 0.

H 值不仅仅通过节点的介数值考虑了节点的位置,而且还将节点与其他节点的连接难易程度考虑其中,通过实验验证,该值更能全面地描述网络中节点的重要性.

当数据经过Ⅱ类节点时,如果将数据存储在 H 值最大的节点中,可以使该数据为更多的客户端节点请求服务. 但是,一旦当网络确定时,网络中节点的位置便固定了,其 H 值也将会被固定,因此,H 值大的节点将产生很大的负载,而 H 值相对较小的节点却显得空闲. 这样一来将会导致 H 值大的节点中的数据频繁地被替换,致使请求的网内命中率大大降低,而其他Ⅱ类节点的资源得不到利用.

针对以上问题,本文给出了较好的解决方法. 由于每个客户端节点对数据的"喜好"往往不同,并且客户端对数据的请求具有实时性,因此,不同客户端对数据请求往往不同以及同一客户端在不同时刻对数据请求也不相同,这样将会使同一数据在不同网内节点(主要考虑Ⅱ类节点)表现出不同的流行度. 本文针对具体数据实时地判断数据所经过的节点哪一个对当前数据的需求较大,并将该需求与节点的 H 值相结合,从而使数据能够动态地存储在不同的节点上,而并非某一 H 值较大节点. 由于数据的请求频率能够很好地度量一个数据的流行度,本文将数据的请求频率看作节点对此数据的需求. 因此,给出节点 v_i 关于数据 q 的需求 D_{iq} 为

$$D_{iq}=\frac{f_{iq}}{\sum\limits_{c=1}^{m}f_{ic}},\tag{4}$$

其中,f_{iq} 表示节点 v_i 中数据 q 的请求频率,并设节点 v_i 的大小为 m(即能够存储 m 个数据块).

式(4)虽然能够很好地表示出具体数据在某一节点的需求,但很明显式(4)不具备实时性. 如果某一数据在一段时间内得到了大量的请求,但是在其后的很长一段时间内都没有请求该数据,但由于前一段时间的高频率请求导致其有较大的 D 值,从而不能实时地反映出数据当时的情况. 因此,为了减少该问题带来的影响,现给出式(5)表示节点 v_i 对数据 q 需求的实时表达式.

$$M_{iq}(t_{\mathrm{gap}})=\frac{f_{iq}}{\sum\limits_{c=1}^{m}f_{ic}}\mathrm{e}^{t_{\mathrm{gap}}},\tag{5}$$

$$t_{\mathrm{gap}}=\frac{1}{t_{\mathrm{current}}-t_{\mathrm{old}}},\tag{6}$$

其中,t_{gap} 表示 q 数据 2 次请求的时间间隔,t_{current} 表示本次请求到达节点的时刻,t_{old} 表示该请求上次到达该节点的请求时刻.

由式(3)与式(5)可得节点 v_i 对数据 q 的重要性的实时度量为

$$W_{iq}=M_{iq}(t_{\mathrm{gap}})\times H(i)=\left[\frac{f_{iq}}{\sum\limits_{c=1}^{m}f_{ic}}\mathrm{e}^{t_{\mathrm{gap}}}\right]\times\left[B(i)+\sum_{j=1}^{n}\frac{1}{C(j)}B(j)\right],\tag{7}$$

$$v_{kq}=\max_{i\in V}W_{iq}.\tag{8}$$

当请求包经过某一Ⅱ类节点时,将得此节点针对当前数据的 W 值,然后将该值添加到请求包中,当请求包到达下一节点时得到当前数据在此节点的 W 值,并将其与请求包中的 W 值进行比较,将较大者加入到请求包中,以此类推. 当数据包返回时根据请求包传给它的 W 值找到对应的节点,将数据缓存在该节点(即式(8)中的 v_{kq})中即可.

2.2 节完整地描述了 BNC 策略在Ⅰ类节点与Ⅱ类节点的缓存机制的实现. 当请求在Ⅰ类节点就被满足时(即请求包跳数小于 3 时被满足),这时数据在返回客户端的过程中只需要按照数据在Ⅰ类节

点中的存储策略缓存数据. 反之,数据包返回的过程中,将经过Ⅰ类节点与Ⅱ类节点这2类节点,此时,数据将在经过的这2类节点中分别存储1次,其具体缓存策略严格按照Ⅰ类节点与Ⅱ类节点的存储策略执行. 为了更清晰地说明BNC缓存策略的实现过程,伪代码1给出了整个BNC策略实现的伪代码. 为了将Ⅰ类节点与Ⅱ类节点作为一个整体处理,伪代码不以节点类型的形式给出,而是分兴趣包与数据包2部分给出.

伪代码1. BNC策略实现伪代码.

1) 兴趣包处理伪代码:

假设当前节点为 v,用户请求的数据为 q;
if (节点 v 没有存储请求对应数据,并且当前节点到客户端的距离 $hop<3$)
  转发请求到下一节点;
end if
if (节点 v 没有存储请求对应数据,并且当前节点到客户端的距离 $hop>3$)
  得到节点 v 的 id;
  for each ($i=0$ to n) /* n 为节点个数 */
    得到节点 v 对应的 $H(v)$;
  end for
  for each (f_{vi}, $i=0$ to m)
    得到数据 q 在节点 v 的 $M_{vq}(t_{\text{tag}})$;
  end for
  得到 $W_{vq}=H(v)\times M_{vq}(t_{\text{tag}})$;
  得到请求包中携带的上游节点的 W_{interest} 值;
  if ($W_{\text{interest}}<W_{vq}$)
    用 W_{vq} 替换请求包中的 W_{interest} 值;
    转发请求到下一跳节点;
  end if
end if
if (节点 v 存储了请求对应数据,且节点 v 到客户端的距离 $hop>3$)
  添加 W_{interest} 到数据包;
  添加缓存次数 $copyNum=2$ 到数据包;
  返回数据包;
end if
if (节点 v 存储了请求对应数据,且节点 v 到客户端的距离 $1<hop\leqslant 3$)
  从存储列表中得到数据的位置 $rate$; /* $rate$ 为从队头开始数据在队列中的位置与队列中的数据个数的比值 */
  if ($rate\leqslant 0.7$)
    添加 $copyNum=0$ 到数据包中;
  else
    添加 $copyNum=1$ 到数据包中;
  end if
  返回数据包;
end if

2) 数据包处理伪代码

从数据包中获得本节点到客户端的距离 hop 与缓存次数 $copyNum$;
if ($hop>3$)
  得到本节点的 W_{vq};
  得到数据包中携带的 W 值;
  if ($W=W_{vq}$)
    将数据缓存到节点的CS中;
    $copyNum=copyNum-1$;
    添加 $copyNum$ 到数据包中;
    转发数据包到下一节点;
  end if
end if
else if (($hop<3$ && $copyNum=1$) || ($copyNum=1$ && $hop=3$))
  缓存数据到节点的CS中;
  $copyNum=copyNum-1$;
  添加 $copyNum$ 到数据包中;
  转发数据包到下一节点;
end else if
else
  转发数据包到下一节点;
end else

3 仿真实验

本文研究的是在NDN中怎样放置数据(数据存储在哪些节点上),而不涉及节点中数据的替换策略. LRU是一个被普遍使用的数据替换策略,因此,在实验中所有的数据替换策略都使用LRU策略. 为了更好地体现出BNC策略的优越性,本文将NDN中传统的处处缓存策略(LCE)作为对比算法之一. 另外选取2种新近提出的数据缓存策略Betw与ProbCache作为对比算法. 其中,Betw是一种基于介数的缓存机制;ProbCache是一种基于链路节点缓存能力与节点所在位置的概率存储机制.

本文为了更全面地衡量 BNC 策略性能的优越性,将参考多个参数对 BNC 与 LCE,Betw,ProbCache 数据存储策略进行对比.其中包括请求网内平均命中率 β、平均请求跳数 γ、平均命中时延 δ.同时还分别针对缓存相对大小(relative cache size)与 Zipf 参数(Zipf parameter α)[24]分别进行分析.

3.1　性能参数

NDN 与传统网络架构有所不同,由于 NDN 中每个节点都具有缓存数据的功能,因此,请求很有可能在中间节点就被满足而无需到达 OCS 节点,这样不仅可以减少用户请求数据的时延与命中跳数,还能使 OCS 节点的负载减小.因此,在度量 NDN 性能的参数上也与传统网络有所区别.为了更好地理解实验结果,现将本文用到的性能指标定义给出如下:

1) 请求网内平均命中率 β.请求网内命中率能够很好地衡量对 NDN 网内节点缓存的利用效率.β 值越高说明请求的网内命中率也就越高,缓存的利用效率也就越高.

2) 平均请求跳数 γ.平均请求跳数用于衡量数据请求的平均距离(请求数据所经过的节点数),一个好的替换策略应该使请求在较近的范围内被满足.因此,γ 值越小说明数据被满足的平均距离越小,请求的效率越高.

3) 平均命中时延 δ.这应该是最直接表示客户端满意度的参数,如果在没有其他条件影响的情况下,跳数越小时延应该越小,但时延往往受带宽以及服务器位置的影响.δ 值越小表示用户获取数据的平均时间越短,这样客户度的满意度越高.

3.2　仿真环境与参数配置

本实验是在 Ubuntu 操作系统上基于 NS3 的 ndnSIM[25]仿真平台进行仿真实验.每个节点的容量大小相等为 C_i,每个内容的大小为 S_m(本文每个内容大小相等为 1 024 KB),每个节点缓存的内容满足关系:

$$\sum_{m=1}^{n} S_m \leqslant C_i. \tag{9}$$

在实验中分别取网内节点 CS 的总容量为数据总容量的 10%～50%进行实验.为了考虑请求热度变化对 BNC 的影响,本实验取 Zipf 参数 α 在 0.2～1.0 的变化范围进行实验.

为了更好地验证本文提出的 BNC 策略的可扩展性,本文分别在 ARPA(advanced research project agency)网与随机网络拓扑(random topology, RT)上进行实验.其中,随机拓扑通过 BRITE 拓扑生成器生成,2 种拓扑的具体信息如表 1 所示:

Table 1　Information of Network Topology

表 1　网络拓扑信息

Topology	Nodes Number	Max Degree	Min Degree	Links Number	Clients Number	Servers Number
ARPA	21	4	2	27	5	5
RT	68	11	2	136	10	10

客户端请求到达服从泊松分布,其中,ARPA 网中 λ 为每秒 50 个,在随机拓扑中 λ 为每秒 100 个.下面分别在 2 种网络拓扑中进行实验,并对上述给出的相应指标进行统计分析.

3.3　ARPA 网络拓扑实验结果

3.3.1　缓存大小对策略的影响

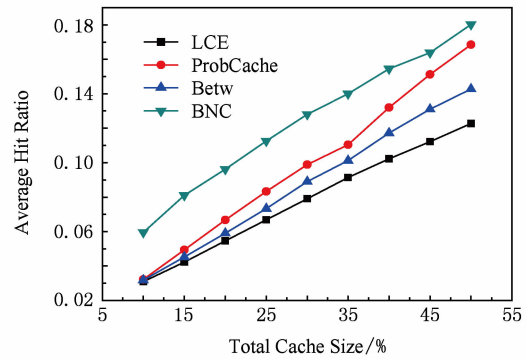
一般情况下,网络中节点的缓存大小将远远小于内容的数量.本文为了考虑随着网内节点缓存大小的变化对 BNC 策略的影响,分别取节点总容量为数据总量的 10%～50%之间的 9 个位置作为实验点,从而观察 BNC 随着网内缓存总容量的变化对其性能的影响.

如图 3(a)所示,随着网内缓存总容量的增加,4 种策略的命中率都有显著的提高.因为,当网内缓存总容量增加时,将有更多的数据被缓存在网内节点,因此,更多的请求被满足,从而使其命中率增加.相对于其他 3 种缓存策略而言,本文提出的 BNC 策略有更高的命中率.BNC 策略最大命中率要比传统 LCE 策略最大命中率高 9%左右.与新近提出的 Betw 策略与 ProbCache 策略相比,BNC 命中率也明显较高.

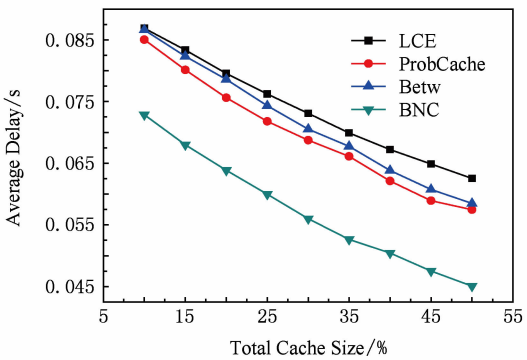
高的命中率不一定与较低的命中跳数和时延相关.例如一个请求在 2 个不同节点请求到相应数据所需要的跳数往往是不一样的;但跳数的远近与时延的大小往往相关联.然而,时延也受到带宽等的影响,如图 3(b)与图 3(c)所示.无论是在请求时延还是请求跳数上 BNC 策略都明显优于其他 3 种策略,BNC 策略与 LCE 策略相比平均跳数最大相差 2 跳左右,时延相差达到了 0.0175 s 左右.BNC 策略与 Betw,ProbCache 策略相比在时延与跳数性能上也有明显优势.

3.3.2　Zipf 参数 α 对策略的影响

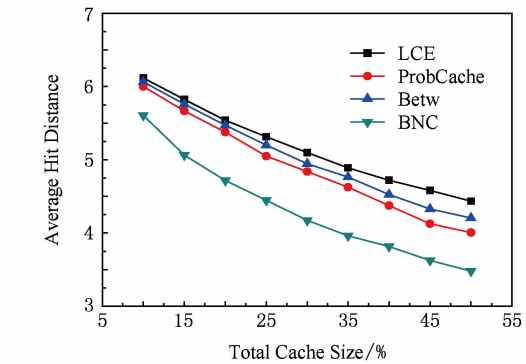
由于用户请求服从 Zipf 分布已经成为了大家的共识,当 Zipf 参数 α 越大时,表示用户请求的数据越集中.即随着 Zipf 参数 α 的增大,具有高请求



(a) Average hit ratio changing with total cache size



(b) Average delay changing with total cache size



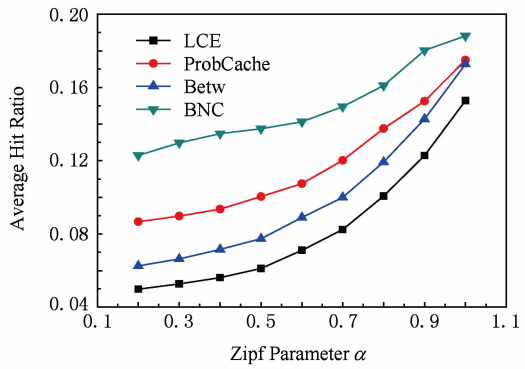
(c) Average hit distance changing with total cache size

Fig. 3 The impact of total cache capacity in ARPA network topology on schemes.

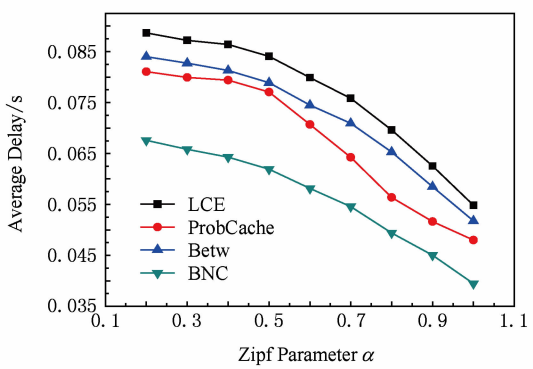
图 3 ARPA 网络拓扑中存储总容量对策略的影响

频率的数据种类将减少. 在本实验中,为了研究用户请求热度变化对缓存策略的影响,分别对 Zipf 参数 α 在 0.2～1.0 的取值范围的情况进行实验.

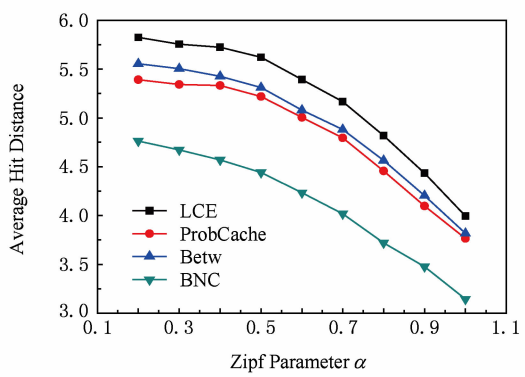
如图 4(a)所示,当 Zipf 参数 α 增加时,请求命中率都相应增加,因为随着 Zipf 参数 α 的增加用户请求的数据类型越来越集中,因此,节点缓存的数据越来越容易满足客户端的请求. 由图 4(a)可以看出,BNC 策略的命中率相对于其他 3 种策略较高;在 Zipf 参数 $\alpha=0.2$ 时 BNC 策略与 LCE 策略相比命中率差值达到了 7%左右,此时,LCE 策略命中率却不到 5%,由此可见,BNC 策略在 Zipf 参数 α 较小时性能远远好于传统 LCE 策略;随着 Zipf 参数 α 的增大,3 种策略之间的差值在逐渐减小,但是 BNC 策略也明显优于其他 3 种策略.



(a) Average hit ratio changing with α



(b) Average delay changing with α



(c) Average hit distance changing with α

Fig. 4 The impact of different Zipf parameter α in ARPA network topology on schemes.

图 4 ARPA 网络拓扑中 Zipf 参数 α 不同时对策略的影响

如图 4(b)与图 4(c)所示,随着 Zipf 参数 α 的增加,用户数据请求时延与跳数都随之减小. 由图 4 可以看出,BNC 策略下用户请求数据的平均时延与跳数远远小于 Betw, ProbCache, LCE 策略. Zipf 参数 $\alpha=0.2$ 时 BNC 策略与 LCE 策略相比时延差值达到 0.02 s 左右,此时,请求跳数差值达到 1.1 跳左

右;在整个取值区间内,BNC 策略在请求时延与跳数性能上优于其他 3 种策略.

3.4　随机网络拓扑(RT)实验结果

为了验证本文提出的策略的可扩展性,因此,提供了在随机拓扑中的仿真结果,从仿真结果来看,本文提出的 BNC 策略性能明显优于其他 3 种对比策略.

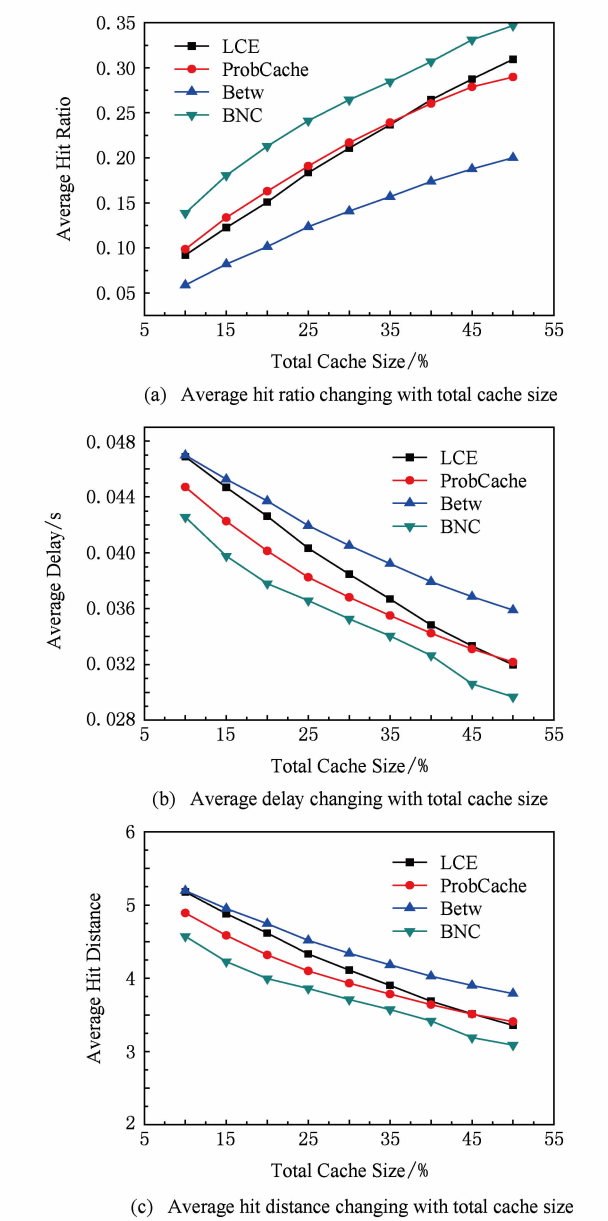
3.4.1　缓存大小对策略的影响

由图 5(a)可以看出,在随机拓扑中随着缓存的增加,4 种策略的命中率都大大提高;但是,本文提出的 BNC 策略性能明显优于其他 3 种策略,当缓存容量为总容量的 50%时,BNC 策略的命中率可以达到 30%以上.但由于随机拓扑节点的差异性巨大,因此,从图 5(a)可以看出 Betw 策略的性能不佳.

在图 5(b)与图 5(c)可以看出,本文提出的 BNC 策略在平均请求时延与平均请求跳数上性能优于 Betw,ProbCache,LCE 策略.

3.4.2　Zipf 参数 α 对策略的影响

由图 6 可以看出,BNC 策略在平均请求命中率、时延、跳数等性能上明显优于 Betw,ProbCache,LCE 三种策略.在图 6(a)中,Zipf 参数 α 增加时,4 种策略的请求命中率都相应增加;但是,本文所提



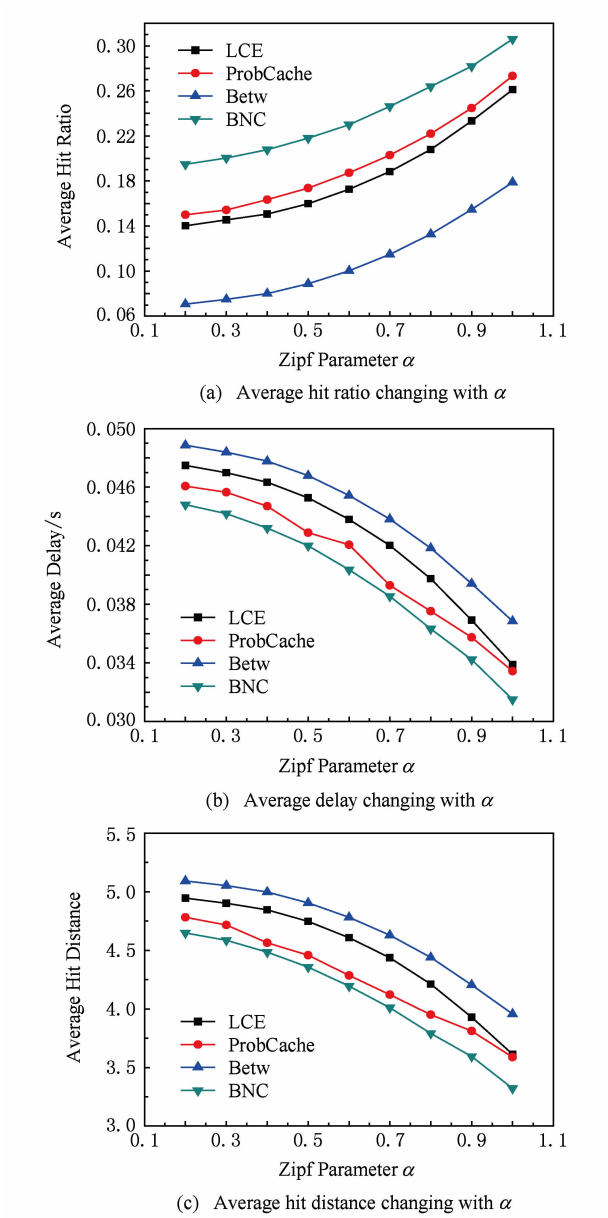
(a)　Average hit ratio changing with total cache size

(b)　Average delay changing with total cache size

(c)　Average hit distance changing with total cache size

Fig. 5　The impact of total cache capacity in Random network topology on schemes.

图 5　随机网络拓扑中存储总容量对策略的影响



(a)　Average hit ratio changing with α

(b)　Average delay changing with α

(c)　Average hit distance changing with α

Fig. 6　The impact of Zipf parameter α in Random network topology on schemes.

图 6　随机网络拓扑中 Zipf 参数 α 对策略的影响

出的策略命中率远高于其他 3 种策略.本文提出的命中率与对比算法的命中率最大差值可达到 15 个百分点.

在图 6(b)与图 6(c)可以看出,BNC 策略在用户体验上也明显优于其他 3 种对比策略;其用户请求时延与数据传输跳数都较其他 3 种策略小,这样将提高用户体验.

4 结束语

为了解决数据包返回客户端过程中的数据缓存位置的问题,本文将数据返回客户端所经节点进行分类处理,由于不同类型节点的作用对象不同,本文分别采用不同的缓存方式,提出了一种全新的数据存储策略 BNC.该策略大大减少了网内节点的冗余数据,同时在满足当前客户端节点的同时还满足了其他客户端节点的需求.通过与传统的 LCE 策略以及新近提出的 Betw,ProbCache 策略相比,本文提出的 BNC 策略在增加请求命中率的同时也减少了用户的请求时延与跳数,从而使整个网络的性能得到提升.

参 考 文 献

[1] Feldmann A. Internet clean-slate design: What and why? [J]. ACM SIGCOMM Computer Communication Review, 2007, 37(3): 59-64

[2] Cheriton D R, Gritter M. TRIAD: A scalable deployable NAT-based Internet architecture [R]. Palo Alto, CA: Stanford University, 2000

[3] Koponen T, Chawla M, Chun B G, et al. A data-oriented (and beyond) network architecture [J]. ACM SIGCOMM Computer Communication Review, 2007, 37(4): 181-192

[4] Han D, Anand A, Dogar F, et al. XIA: Efficient support for evolvable inter-networking [C] //Proc of the 9th USENIX Conf on Networked Systems Design and Implementation. Berkeley, CA: USENIX Association, 2012: 23-23

[5] The FP7 4WARD Project [OL]. European Commission: FP7, 2008 [2014-10-10]. http://www.4ward-project.eu

[6] Scalable and Adaptive Internet Solutions (SAIL) [OL]. European Commission: FP7, 2010 [2014-10-10]. http://www.sail-project.eu

[7] Content mediator architecture for content-aware networks (COMET) [OL]. Madrid: EU ICT, 2010 [2014-10-10]. http://www.comet-project.org

[8] Jacobson V, Smetters D K, Thornton J D, et al. Networking named content [C] //Proc of the 5th ACM Int Conf on Emerging Networking Experiments and Technologies (CoNEXT'09). New York: ACM, 2009: 1-12

[9] Fayazbakhsh S K, Lin Y, Tootoonchian A. Less pain, most of the gain: Incrementally deployable ICN [J]. ACM SIGCOMM Computer Communication Review, 2013, 43(3): 147-158

[10] Wang Yonggong, Li Zhenyu, Tyson Gareth, et al. Optimal cache allocation for content centric networking [C] //Proc of the 21st IEEE Int Conf on Network Protocols (ICNP 2013). Piscataway, NJ: IEEE, 2013: 1-10

[11] Wang Jingyuan, Li Chao, Xiong Zhang, et al. Survey of data centric smart city [J]. Journal of Computer Research and Development, 2014, 51(2): 239-259 (in Chinese)
(王静远, 李超, 熊璋, 等. 以数据为中心的智慧城市研究综述[J]. 计算机研究与发展, 2014, 51(2): 239-259)

[12] Ghodsi A, Shenker S, Koponen T, et al. Information centric networking: Seeing the forest for the trees [C] //Proc of the 10th ACM Workshop on Hot Topics in Networks (HotNets-X). New York: ACM, 2011: 1-12

[13] Jacobson V, Smetters D K, Thornton J D, et al. Networking named content [J]. Communications of the ACM, 2012, 55(1): 117-124

[14] Xie Gaogang, Zhang Yujun, Li Zhenyu, et al. A survey on future Internet architecture [J]. Chinese Journal of Computers, 2012, 35(6): 1109-1119 (in Chinese)
(谢高岗, 张玉军, 李振宇, 等. 未来互联网体系结构研究综述[J]. 计算机学报, 2012, 35(6): 1109-1119)

[15] Ahlgren B, Dannewitz C, imbrenda C, et al. A survey of information-centric networking [J]. IEEE Communications Magazine, 2012, 50(7): 26-36

[16] Xylomenos G, Ververidis C, Siris V, et al. A survey of information-centric networking research [J]. IEEE Communications Surveys & Tutorials, 2013, 16(2): 1024-1049

[17] Laoutaris N, Syntila S, Stavrakakis I. Meta algorithms for hierarchical Web caches [C] //Proc of the 23rd IEEE Int Performance, Computing, and Communications Conf. Piscataway, NJ: IEEE, 2004: 445-452

[18] Laoutaris N, Che H, Stavrakakis I. The LCD interconnection of LRU caches and its analysis [J]. Performance Evaluation, 2006, 63(7): 609-634

[19] Liu Waixi, Yu Shunzheng, Hu Xiao, et al. Selective caching in content centric networking [J]. Chinese Journal of Computers, 2014, 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, 2014, 37(2): 275-288)

[20] Chai Weikoong, He Diliang, Ioannis P, et al. Cache "Less for More" in information centric networks [C] //Proc of the 11th Int IFIP TC6 Networking Conf (NETWORKING 2012). Berlin: Springer, 2012: 27-40

[21] Li J, Wu H, Liu B, et al. Popularity-driven coordinated caching in named data networking [C] //Proc of the 8th ACM/IEEE Symp on Architectures for Networking and Communications Systems. New York: ACM, 2012: 15-26

[22] Psaras I, Chai W K, Pavlou G. Probabilistic in-network caching for information centric networks [C] //Proc of the 2nd ACM SIGCOMM Information-Centric Networking Workshop (ICN 2012). New York: ACM, 2012: 55-60

[23] Zhang Xiaojuan, Wang Xufeng. Evaluation formula for communication network node importance [J]. Journal of Northeastern University, 2014, 35(5): 663-666 (in Chinese)
(张小娟, 王旭峰. 一种通信网络节点重要性计算公式[J]. 东北大学学报, 2014, 35(5): 663-666)

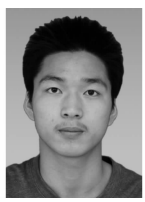
[24] Breslau I, Cao P, Fan I, et al. Web caching and Zipf-like distributions: Evidence and implications [C] //Proc of the 18th Annual Joint Conf of the IEEE Computer and Communications Societies: The Future is Now (INFOCOM'99). Piscataway, NJ: IEEE, 1999: 126-134

[25] Afanasyev A, Moiseenko I, Zhang L. ndnSIM: NDN simulator for NS-3, NDN-0005 [R]. Los Angeles, CA: University of California, Los Angeles, 2012

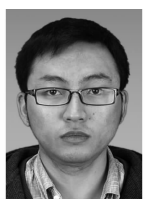
Huang Sheng, born in 1974. Received his MS degree from Huazhong University of Science & Technology and his PhD degree from Chongqing University in 2003 and 2008, respectively. Member of China Computer Federation. His main research interests include optical communication system, channel coding and named data networking.

Teng Mingnian, born in 1991. Master candidate. Received his bachelor's degree from Qiqihar University in 2013. His current research interests include data packet caching in named data networking.

Wu Zhen, born in 1990. Master candidate. Received his bachelor's degree from Hunan First Normal University in 2013. His current research interests include named data networking.



Xu Jianghua, born in 1990. Master candidate. Received his bachelor's degree from Taiyuan University of Science and Technology in 2013. His current research interests include IP lookup and name lookup in CCN.



Ji Ruijun, born in 1990. Master candidate. Received his bachelor's degree from Chongqing University of Posts and Telecommunications in 2013. His current research interests include LT code.