Journal of Computer Applications
计算机应用,2014,34(12):3369 -3372
ISSN 1001-9081
CODEN JYIIDU
2014-12-10
http://www.joca.cn

文章编号:1001-9081(2014)12-3369 -04
doi:10.11772/j.issn.1001-9081.2014.12.3369

命名数据网络中基于数据请求代价与流行度的动态替换策略

黄　胜,滕明埝*,陈胜蓝,刘焕淋,向劲松
(重庆邮电大学 光纤通信技术重点实验室,重庆 400065)
(*通信作者电子邮箱 tengmingnianzbb@163.com)

摘　要:针对怎样高效地对命名数据网络(NDN)缓存中的数据进行替换的问题,提出了一种综合考虑数据流行度与数据请求代价的数据替换策略。该策略根据数据的请求时间间隔动态地分配数据流行度因子与数据请求代价因子的比重,使节点缓存高流行度与高请求代价的数据。当用户下次请求数据时能够从本节点获取,降低数据请求的响应时间并减少链路拥塞。仿真结果表明,本策略能够有效提高网内存储命中率,降低用户获取数据的时间以及缩短用户获取数据的距离。

关键词:命名数据网络;替换策略;未来互联网
中图分类号:TP393　　**文献标志码**:A

Dynamical replacement policy based on cost and popularity in named data networking

HUANG Sheng, TENG Mingnian*, CHEN Shenglan, LIU Huanlin, XIANG Jinsong
(*Key Laboratory of Optical Fiber Communication Technology, Chongqing University of Posts and Telecommunications, Chongqing* 400065, *China*)

Abstract: In view of the problem that data for Named Data Networking (NDN) cache is replaced efficiently, a new replacement policy that considered popularity and request cost of data was proposed in this paper. It dynamically allocated proportion of popularity factor and request cost factor according to the interval time between the two requests of the same data. Therefore, nodes would cache data with high popularity and request cost. Users could get data from local node when requesting data next time, so it could reduce the response time of data request and reduce link congestion. The simulation results show that the proposed replacement policy can efficiently improve the in-network hit rate, reduce the delay and distance for users to fetch data.

Key words: Named Data Networking (NDN); replacement policy; future Internet

0　引言

随着互联网的迅速发展,网络中的数据呈爆炸式增长,用户已经不再关心所请求数据的位置在哪儿,而关心的是所请求的数据是什么,因此,最初以提供端到端连接为目的的基于"客户机-服务器(C-S)"模式的互联网已经不再适合当前用户对网络的需求[1],由此引起了人们对未来互联网架构的研究热潮,命名数据网络(Named Data Networking,NDN)就是众多未来互联网架构中最具代表性的网络架构之一[2-4]。

缓存替换策略(Cache Replacement Policy,CRP)是 NDN 的研究热点之一。在 NDN 中,每个节点都具有内容存储库(Content Store,CS),用于长时间缓存经过的数据,缓存后的数据可以满足后续与数据对应的请求。这也是 NDN 不同于 IP 网络的最大特性。与海量的数据相比,节点的 CS 容量相当有限,如何对 CS 中的数据进行替换,尽可能地减小内容请求失败的概率(Request Miss Probability,RMP)以及降低数据的请求时延(Data Request Delay),已经成为影响 NDN 性能的关键因素。

最近最少使用策略(Least Recently Used,LRU)、最不经常使用策略(Least Frequently Used,LFU)、先进先出策略(First In First Out,FIFO)是 NDN 中常见的几种缓存替换策略,其中,LRU 策略是将最近最少使用的数据替换,LFU 策略是将使用频率最少的数据替换,FIFO 策略是根据数据请求的先后时间将最先请求的数据替换,这三种策略都有一个共同的缺陷,即考虑的替换因素太过单一,没有考虑数据的请求代价等影响因素[5-7]。由文献[7]可知 LRU 策略与 FIFO 策略都不能实时反映数据流行度。LFU 由于其静态特性,往往不能及时地反映当前数据请求的流行度,即如果某数据在很短一段时间内被大量请求,使该数据拥有较大的请求频率,在一段时间后该数据的请求量急剧下降,但由于前面的高频率请求使该数据获得了较大的权重,因此该数据即使当前请求频率很低也不能及时地将其替换,从而长期占用内存空间,使当前具有高流行度或高请求代价的数据不能被缓存。为了考虑数据请求代价对缓存的影响,Wang 等基于 GreedyDual-Size 策略[8]提出一种改进型的 Hetero[9]策略,该策略将节点获取数据的跳数作为代价,并根据代价将数据的权重设为 $L+C_p$,C_p

收稿日期:2014-06-27;**修回日期**:2014-08-09。　**基金项目**:国家自然科学基金资助项目(61371096,61275077);重庆市自然科学基金资助项目(cstc2013jcyjA40052);重庆市教委科学技术研究项目(KJ130515)。

作者简介:黄胜(1974 -),男,湖北英山人,教授,博士,主要研究方向:未来互联网体系结构、计算机网络、光网络;　滕明埝(1991 -),男,重庆人,硕士研究生,主要研究方向:未来互联网数据存储策略;　陈胜蓝(1986 -),男,重庆人,硕士研究生,主要研究方向:未来互联网数据存储策略;　刘焕淋(1970 -),女,重庆人,教授,博士,主要研究方向:宽带网络技术、光纤通信、光交换技术;　向劲松(1975 -),男,重庆人,副教授,博士,主要研究方向:空间光通信。

为节点获取该数据的需要经过的跳数，L 初始化为 0，每次将权重值最小的数据替换[9]。Chen 等[10]根据 NDN 的特点，提出 LUV-Path 策略，该策略将数据的权重函值设为本节点与服务器之间的距离，则距离服务器较远的数据可能具有相对较大的权重值，相对于权重较小的数据，权重值较大的数据更易被节点缓存。以上策略虽然都考虑了数据请求的代价，能够在一定程度上降低数据请求时的代价，但这些策略都没有考虑数据的流行度，而且这些策略不能实时反映数据当前的情况。

基于以上问题，本文提出了一种基于数据请求代价与数据流行度的动态缓存替换策略。该策略每次根据数据的请求时间差，分别赋予数据请求流行度因子与数据请求代价消耗因子不同的权重系数。本策略根据数据当时的情况，结合数据流行度与数据请求代价，实时地调整数据的权重，使数据的存储更合理。仿真数据证明，该策略能够大大提高数据请求的网内命中率并减小用户请求数据的时延与数据请求距离。

1 NDN 工作原理

NDN 是一种信息中心网络（Information-Centric Networking，ICN）[11-12]，该结构不再关心数据的存储位置，只关心数据本身。NDN 中的数据不再使用 IP 地址作为数据的标识，而是以数据的名称作为标识。NDN 中存在两种数据包类型：请求包（Interest Packet）和数据包（Data Packet）。用户发送包含数据名称的请求包，该请求包被路由到含有对应数据的邻近节点或服务器节点；然后，将找到的数据包沿着请求包的反向路径传送给数据请求者。

如图 1 所示，当请求包到达时，节点首先查找 CS 中是否有与该请求对应的数据包，如果有就将数据传回给请求者，否则节点会查询未决请求表（Pending Interest Table，PIT）中是否含有与该请求对应的条目，如果存在就将请求进入接口添加到对应条目的接口列表中，然后将请求抛弃，否则新建一个 PIT 条目并查询节点的转发信息库（Forwarding Information Base，FIB），将请求转发到下一跳节点。

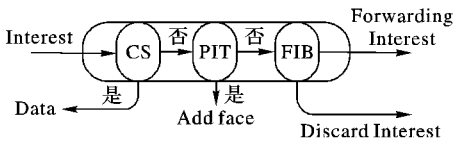

图 1 请求包转发过程

如图 2 所示，数据包到达时，节点会查询 PIT，如果 PIT 中存在与数据对应的请求条目就将数据从该条目的接口列表中转发出去，并根据相应的存储策略将其存储在节点的 CS 中。NDN 节点的 CS 能够长期存储数据包，与当前互联网“C-S”模式不同，请求包在到达服务器之前，就可能在某个中间节点找到匹配的数据包。如何高效地替换存储的数据成为了 NDN 研究领域中的热点课题之一[2-4]。

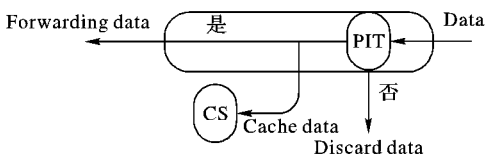

图 2 数据包转发过程

2 基于请求代价与流行度的动态替换策略

2.1 CPRP 策略基本思想

NDN 中的每个节点都具有 CS，其缓存的数据可以用来满足与这些数据对应的请求。好的数据替换策略可以使 CS 存储更有利于网络整体性能的数据，提高请求的网内命中率，减小服务器节点的压力，同时也能够使用户请求数据的时延大大降低。基于以上分析，本文提出的基于数据请求代价与流行度的动态替换策略（Cost and Popularity based Replacement Policy，CPRP）替换策略利用两次数据请求的时间间隔，动态地分配数据流行度（数据请求频率）与数据请求代价（当前节点与提供数据所对应的服务器节点距离，即跳数）的权重，使数据获得的权重值更符合数据当时的情况。

由于用户对数据的需求是不断变化的，动态地反映数据当时的情况十分有必要，CPRP 将相邻两次数据请求的时间间隔作为权重值分配依据。通过该时间间隔可以很好地评估数据的流行程度，即如果两次数据的请求时间间隔很短，那么，表示该数据在当前时刻的请求频率较高，这时将数据流行度因子作为影响数据权重值的重要因子，而数据请求代价因子作为次要因子；否则，将数据请求代价作为重要因子，数据请求频率作为次要因子。通过这种方式就可以使数据的权重值实时地得到调整，更符合当前数据的情况，从而使得缓存的数据更能够满足当时用户的需求，而不是以前的流行数据占据大量缓存空间使现有高请求数据得不到缓存，致使用户请求数据时都需将请求发送到服务器节点，从而产生额外的网络资源消耗，并且该策略在考虑数据流行度的情况下还充分考虑了数据请求的代价，这样不仅能够对高流行度的数据进行及时缓存，对高请求代价的数据也能够很好地存储，大大减少了网内的数据流量，在提高网络数据请求命中率的同时还能够减少用户请求数据的时延与距离，从而使整个网络的性能大大提高。

2.2 CPRP 模型分析

为了反映出对数据权重值更新的实时性，本策略通过对数据的请求时间间隔长短实时改变权重因子的比重。在每个节点添加一个记录数据上次请求的时间 T_{old}，当数据请求再次到达时用当前时间 $T_{current}$ 减去对应数据上次被请求的时间 T_{old}，得到数据两次请求的时间间隔 $T_{interval}$：

$$T_{interval} = T_{current} - T_{old} \tag{1}$$

当数据到达时，节点根据数据的名称得到该数据的请求频率 F_{data}，同时提取数据包中记录从服务器到达本节点的跳数对应的字段，得到本节点距离服务器节点的距离 H_{hop}，为了体现出本策略在不同时刻数据请求所需代价与数据流行度对数据权重值的影响不同，本策略将这两个因子做加法运算，根据 $T_{interval}$ 分别赋予这两个因子不同的权重值为 $W_{frequency}$ 和 W_{hop}，分别将 F_{data} 与 H_{hop} 归一化处理如下：

$$F_{data\text{-}normalization} = F_{data} / F_{max} \tag{2}$$

$$H_{hop\text{-}nomalization} = H_{hop} / H_{max} \tag{3}$$

由式(2)、(3)以及 $W_{frequency}$ 和 W_{hop} 可以得到数据的权重值 D_{data}：

$$D_{data} = W_{frequency} \cdot F_{data\text{-}normalization} + W_{hop} \cdot H_{hop\text{-}normalization} \tag{4}$$

其中：$W_{frequency} = L$，$W_{hop} = 1 - L$，即：$W_{frequency} + W_{hop} = 1$。

为了体现出对两个权重因子的实时调整的特性，得 L 如

式(5)所示:

$$L = 1/C^{T_{\text{interval}}} \tag{5}$$

通过实验验证当 C 取 e 时 CPRP 特性最佳,如式(6)所示:

$$L = 1/e^{T_{\text{interval}}} \tag{6}$$

由式(6)可得 $W_{\text{frequency}}$ 和 W_{hop} 如下:

$$W_{\text{frequency}} = 1/e^{T_{\text{interval}}} \tag{7}$$

$$W_{\text{hop}} = 1 - (1/e^{T_{\text{interval}}}) \tag{8}$$

由式(7)、(8)可以得出 D_{data}:

$$D_{\text{data}} = (1/e^{T_{\text{interval}}}) \cdot F_{\text{data-normalization}} + (1 - (1/e^{T_{\text{interval}}})) \cdot H_{\text{hop-nomalization}} = (1/e^{T_{\text{interval}}}) \cdot (F_{\text{data}}/F_{\max}) + (1 - (1/e^{T_{\text{interval}}})) \cdot (H_{\text{hop}}/H_{\max}) \tag{9}$$

由式(9)可以得出,当某一数据的两次请求时间间隔 T_{interval} 越短时 $W_{\text{frequency}}$ 将获得较大的比重,W_{hop} 相对而言比重会减小,这与本策略的初衷相符。由文献[7]可知短时间内如果某数据被连续请求,那么,该数据的请求时间间隔 T_{interval} 很小,说明用户对该数据的需求量很大,即该数据的流行度很高,这时应该将数据的流行度因子作为影响数据权重值的主要因子。由式(7)、(8)可得当 T_{interval} 很小时 $W_{\text{frequency}}$ 远远大于 W_{hop}。当然,当数据的两次请求时间间隔 T_{interval} 较大时,说明该数据在当前请求量较少,即流行度降低,在这种情况下将数据的请求代价作为数据权重值的重要影响因子,而数据流行度则作为次要因子。此时,由于 T_{interval} 较大由式(7)、(8)可得 $W_{\text{frequency}}$ 小于 W_{hop}。通过实时地对数据权重值的调整,可以使 CS 根据当时的情况缓存最具价值的数据。由式(9)可以看出,本策略综合考虑了数据的流行度与数据请求时所需代价,不仅能够对高流行度的数据进行缓存,对高请求代价的数据也能够很好地存储,从而丰富了 CS 中数据的多样性,使整个网络中所缓存的数据更加合理,且每次对数据权重值实时地更新使 CS 中缓存的数据更“新鲜”。

每当数据到达时,如果 CS 有足够的缓存,那么,根据相应的缓存策略将数据缓存到 CS 中;否则根据到达的数据算出数据新的权重值,将得出的权重值与 CS 内已有数据权重值进行比较,删除权重值最小的数据。CS 中数据替换过程的伪代码如下。

```
Data k arrived at Node i;
Bool cache = caching_policy(k);
if(! cache)
  forward_data(k);
else
  get F_max and H_max in list;
  get F_data and H_data in packet;
  compute W_frequency and W_hop according to above;
  compute D_data according to W_frequency and W_hop;
  if there is enough space;
    insert data k into CS according to D_data;
  else
    if( D_data > = D_min)
      evict the data with D_min and insert data k into CS;
    else
      cache false;
    end if
  end if
end if
```

3　仿真分析

本文在 Linux (Ubuntu)环境下,采用基于 NS-3 的 ndnSIM[13]进行仿真,仿真拓扑共有 120 个节点,共设置了 4 个客户端节点与 1 个原始服务器节点,分别通过边缘节点进入网络,服务节点提供 2000 个不同的数据,每个数据的大小为 1024 KB,数据包的流行度服从 Zipf 分布,α 为 0.75。请求的频率服从泊松分布 λ = 10 req/s,服务器节点提供与用户请求相应的数据包,网内节点 CS 的总容量为数据总数的10% ~ 50%,在 Prob(p)与 LCE(Leave Copy Everywhere)两种存储模式下,分别比较六种替换策略 LRU、LFU、FIFO、CPRP、LUV-Path、Hetero 的性能,仿真时间为 200 s。

图 3 与图 4 为在 LCE 与 Prob(p)两种替换策略下,网内存储容量不同时,用户请求在网内的平均命中率。该命中率为每一个采样时刻所有网内节点服务的请求数与总的请求数之比。与其他五种策略相比,CPRP 策略提高了请求在网内的命中率。因为,本策略实时地对 CS 中的数据进行替换,使流行数据的流行度与实际情况相吻合,从而在不同时刻能够更好地满足当时用户对数据的需求,因此与 FIFO、LRU、LFU、LUV-Path、Hetero 相比,本策略使网内请求命中率大大增加。

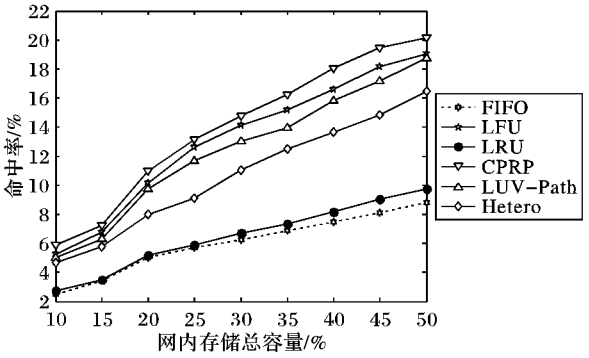

图 3　基于 Prob(p)存储策略的平均命中率

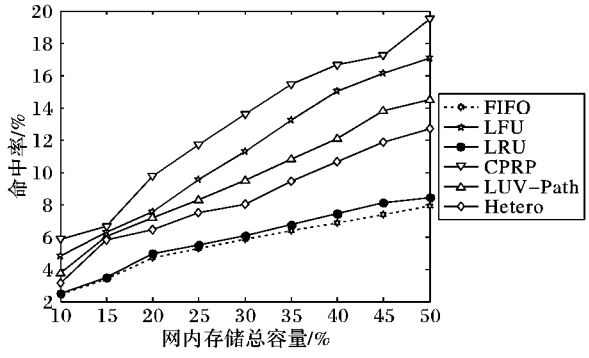

图 4　基于 LCE 存储策略的平均命中率

图 5 与图 6 为在 LCE 与 Prob(p)两种替换策略下,当网内缓存增大时六种替换策略在平均时延方面的性能表现。可以看出在两种缓存策略下,CPRP 替换策略使用户的数据请求平均时延更小。因为,本策略在对 CS 中的数据进行替换时不仅考虑了数据的流行度因子还将数据的请求代价作为重要因素进行考虑,因此,CS 中存储的数据不仅具有高流行度还具有高请求代价特性;从而使用户再次请求高请求代价的数据时能够由网内节点满足,而不需要向服务器节点发起请求,因此,使用户请求数据的时延大大降低。

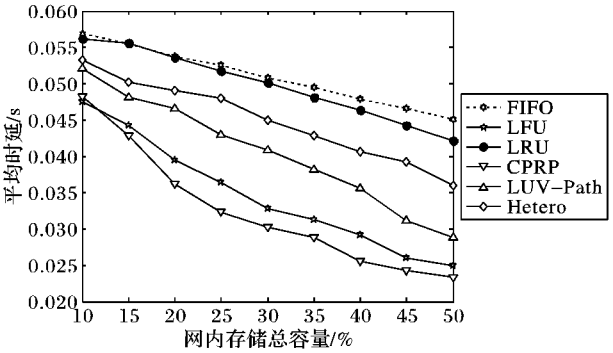

图 5 基于 Prob(p)存储策略用户获取数据的平均时延

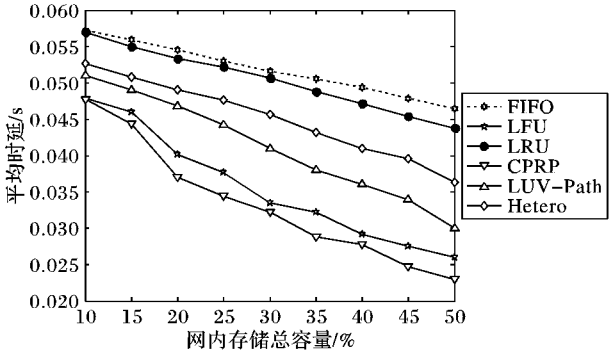

图 6 基于 LCE 存储策略用户获取数据的平均时延

图 7 与图 8 为在 LCE 与 Prob(p)两种替换策略下,当网内缓存增大时,六种替换策略在平均请求跳数方面的性能表现。由图可知,CPRP 策略能够大大减小用户请求数据时的平均跳数。因为,本策略动态地调节了 CS 中缓存的数据,使缓存的数据具有实时性,因此,本策略缓存的数据更“新鲜”,从而使用户请求数据时能够更容易地在网内节点被满足,而不是向服务器节点或更远的网内节点转发请求,所以,CPRP 策略与其他几种策略相比能够大大降低用户请求数据时的平均跳数。

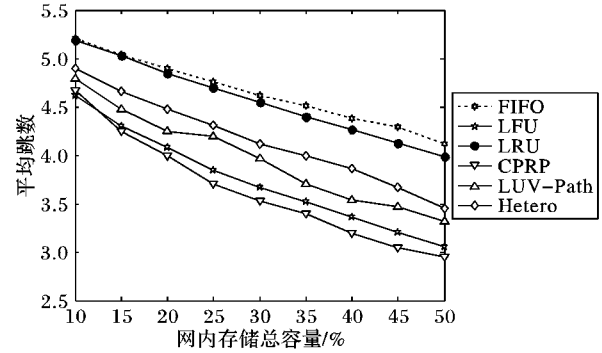

图 7 基于 Prob(p)存储策略用户获取数据的平均跳数

4 结语

NDN 作为未来网架构的突出代表之一,其网内存储功能能够使网络的整体性能得到大大的提升,怎样对网内数据进行合理的替换已经成为了制约 NDN 性能因素之一。为了解决这一问题,本文提出了 CPRP 数据替换策略,该策略根据两次数据请求的时间间隔,合理地分配数据请求代价因子与数据流行度因子的比重,从而使网内数据实时地更新。本策略使网内数据命中率大大提高的同时还减少了用户对数据请求的时延,从而使整个网络的性能得到了改善。

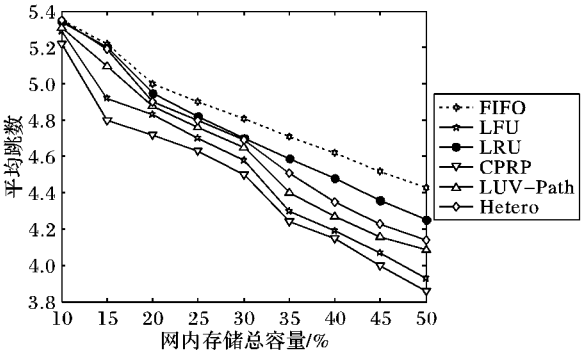

图 8 基于 LCE 存储策略用户获取数据的平均跳数

参考文献:

[1] XIE G, ZHANG Y, LI Z, *et al.* A survey on future Internet architecture[J]. Chinese Journal of Computers, 2012, 35(6): 1109 - 1119.(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.)

[2] 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: ACM, 2009: 1 - 12.

[3] JACOBSON V, SMETTERS D K, THORNTON J D, *et al.* Networking named content[J]. Communications of the ACM, 2012, 55(1): 117 - 124.

[4] ZHANG L, ESTRIN D, BURKE J, *et al.* Named data networking project, NDN-0001[R/OL]. [2014 - 06 - 27]. http://named - data.net/wp - content/uploads/TR001ndn - proj.pdf.

[5] PSARAS I, CLEGG R G, LANDA R, *et al.* Modeling and evaluation of CCN-caching trees[C]// Proceedings of the 10th International IFIP TC 6 Conference on Networking. Berlin: Springer, 2011: 78 - 91.

[6] CAROFIGLIO G, GALLO M, MUSCARIELLO L, *et al.* Modeling data transfer in content-centric networking[C]// Proceedings of the 23rd International Teletraffic Congress. Piscataway: IEEE, 2011: 111 - 118.

[7] CAROFIGLIO G, GALLO M, MUSCARIELLO L. Bandwidth and storage sharing performance in information centric networking[C]// Proceedings of the 2011 ACM SIGCOMM Conference. New York: ACM, 2011: 1 - 6.

[8] CAO P, IRANI S. Cost-aware WWW proxy caching algorithms [C]// Proceedings of the USENIX Symposium on Internet Technologies and Systems. Berkeley: USENIX Association, 1997: 193 - 206.

[9] WANG J M, BENSAOU B. Improving content-centric networks performance with progressive, diversity-load driven caching[C]// Proceedings of the 2012 1st IEEE International Conference on Communications in China. Piscataway: IEEE, 2012: 85 - 90.

[10] CHEN X, FAN Q, YIN H. Caching in information-centric networking: from a content delivery path perspective[C]// Proceedings of the 2013 9th International Conference on Innovations in Information Technology. Piscataway: IEEE, 2013: 48 - 53.

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

[12] XYLOMENOS G, VERVERIDIS C, SIRIS V, *et al.* A survey of information-centric networking research[J]. IEEE Communications Surveys and Tutorials, 2013, 16(2): 1024 - 1049.

[13] AFANASYEV A, MOISEENKO I, ZHANG L. NDNSIM: NDN simulator for NS-3, NDN-0005[R/OL]. [2014 - 06 - 27]. http://named - data.net/publications/techreports/trndnsim/.