软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software, 2022,33(12):4816–4837 [doi: 10.13328/j.cnki.jos.006382]

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

FPTC: 一种信息中心物联网缓存策略*

吴海博[1,2], 许瑶恭[1,2], 李 俊[1,2]

[1](中国科学院 计算机网络信息中心, 北京 100190)

[2](中国科学院大学, 北京 100049)

通信作者: 李俊, E-mail: lijun@cnic.cn

摘 要: 信息中心网络(information-centric networking, ICN)作为一种新型未来互联网体系结构应运而生, 并广泛应用于物联网领域, 其中, 缓存技术作为 ICN 的显著特征, 对信息中心物联网的内容传输性能具有重要影响. 由于信息中心物联网具有数据频繁更新、用户对数据新鲜度有严格要求等特性, 致使传统信息中心网络缓存技术面临挑战. 提出一种基于内容流行度和网络拓扑的分布式缓存策略, 同时考虑内容新鲜度, 各缓存节点通过优先缓存流行度较高且较靠近用户的内容, 以最大化缓存效率. 为适应物联网内容的频繁更新, 提出一种基于灰色预测的内容缓存收益预测方法, 便于快速获取新内容的缓存收益值. 同时, 该策略具有较低的时间和空间开销. 仿真实验结果表明: 所提方案相比于传统缓存策略, 能有效提高缓存效率和命中率, 并降低访问延迟, 改善用户体验.

关键词: 信息中心网络; 物联网; 缓存; 新鲜度; 拓扑; 流行度; 预测

中图法分类号: TP393

中文引用格式: 吴海博, 许瑶恭, 李俊. FPTC: 一种信息中心物联网缓存策略. 软件学报, 2022, 33(12): 4816–4837. http://www.jos.org.cn/1000-9825/6382.htm

英文引用格式: Wu HB, Xu YG, Li J. FPTC: An ICN-IoT Caching Scheme. Ruan Jian Xue Bao/Journal of Software, 2022, 33(12): 4816–4837 (in Chinese). http://www.jos.org.cn/1000-9825/6382.htm

FPTC: An ICN-IoT Caching Scheme

WU Hai-Bo[1,2], XU Yao-Gong[1,2], LI Jun[1,2]

[1](Computer Network Information Center, Chinese Academy of Sciences, Beijing 100190, China)

[2](University of Chinese Academy of Sciences, Beijing 100049, China)

Abstract: As a new future Internet architecture, information-centric networking (ICN) emerges as the time goes and is widely used in the field of Internet of Things. Caching technology is an important feature of ICN, and has an important impact on the content transmission performance in ICN-IoT. Due to the characteristics of data updating frequently and the strict requirements for data freshness in ICN-IoT, the traditional network caching technology of ICN is facing challenges. This study proposes a distributed caching strategy based on content popularity and network topology, considering the freshness of content. To maximize the caching efficiency, each caching node first caches the content with higher popularity and closer to users. In order to adapt to the frequent updating of content in IoT, a content caching reward prediction method based on grey prediction is proposed, which is convenient to obtain the caching reward of new content quickly. Simulation results show that, compared with the traditional caching strategy, the proposed scheme can effectively improve the caching efficiency and hit ratio, reduce the access delay, and improve the user experience.

Key words: ICN; IoT; cache; fressness; topology; popularity; prediction

近年来, 物联网(Internet of things, IoT)得到广泛关注和应用, 成为新一代信息技术的重要组成部分. 全球移动通信系统协会 GSMA 公布的报告[1]显示: 截至 2018 年, 全球的物联网设备连接数量达到了 91 亿, 而在

* 基金项目: 国家自然科学基金(61672490, 61602436, 61601443); 中国科学院国际合作重点项目(241711KYSB20180002); 中国科学院青年创新促进会项目

收稿时间: 2020-12-06; 修改时间: 2021-01-28, 2021-03-26; 采用时间: 2021-05-24

2018–2025 年间, 全球物联网连接(蜂窝和非蜂窝)数量将增加两倍, 达到 252 亿.

然而, 由于物联网基于传统 TCP/IP 网络体系架构, 本质上仍是面向主机(host-oriented)或面向位置(location-oriented)的通信, 因此仍面临传统网络安全性、扩展性、多样性等的问题. 为适应当前用户以内容获取为主的应用需求, 同时克服传统 TCP/IP 网络在安全性、移动性、扩展性等方面的弊端, ICN 网络应运而生, 并成为未来网络重要发展方向[2−4]. 针对 ICN 未来互联网体系结构, 研究界提出了许多种架构, 其中代表性的研究主要有 CCN/NDN[5], DONA[6], COMET[7]和 NetInf[8]等. 由于它们之间存在若干共性特征, 为方便起见, 本文相关内容以 NDN[5]为例进行介绍.

由于 ICN 在扩展性、服务质量、安全性、节能、移动性、容错、支持异构等方面具有优势[9], 非常符合物联网的需求, 因此近年来 ICN 和 IoT 的结合已成为大势所趋[10−15]. 特别地, 考虑到缓存对内容传输的重要影响, 信息中心物联网场景下内容缓存的研究已成为研究热点, 得到广泛关注.

物联网作为在互联网基础上进行延伸和扩展的网络, 用来实现任何时间、任何地点, 人与人、人与物、物与物之间的互联互通, 其技术构成主要包括感知与标识技术、网络与通信技术、计算与服务技术和管理与支撑技术这四大体系[16]. 为了使 ICN 与物联网更好地结合, 最大程度地发挥 ICN 缓存的优势, 需要设计一种高效的缓存机制, 使用户能够从距离较近的节点的缓存中获取想要的内容, 而不需要总是从内容原始生产者处获取, 从而降低整个网络的流量以及用户的访问延迟. 在针对信息中心物联网设计的缓存方案中, 现有的缓存策略主要考虑了要缓存内容的某些属性以及打算缓存该内容的节点. 内容属性主要包括了新鲜度、流行度、瞬时性、时间和特定的生产者等; 缓存节点的属性则主要考虑了电量、节点到生产者(或消费者)的距离等. 现有的缓存策略存在缓存中新旧数据更替不够灵活或者考虑因素不够充分等不足. 我们通过分析认为, 物联网数据的新鲜度及其瞬时性(新数据不断产生)是物联网数据的最重要特征, 而对用户而言, 对数据的访问延迟需越小越好. 物联网设备会对周围环境进行采集和传感, 每隔一段时间会产生一个新的内容, 该内容仅代表某个特定时间的值, 因此, 对于不同时刻产生的任何单个内容, 会有不同的值. 对用户而言, 请求某个物联网设备产生的数据时需要考虑内容的正确性, 即是否是所想要的某个时间产生的内容. Quevedo 等人[17]认为, 物联网环境中的用户更倾向于请求最新的内容. 然而, 用户也可能请求一些较旧的内容, 如查询之前某时刻的温度等. 同时, 用户希望尽快获得想要的数据, 这对缓存策略的设计也有一定的要求.

为了更灵活地考虑缓存数据的新旧与用户的访问延迟问题, 本文提出一种基于内容新鲜度和流行度与网络拓扑的分布式缓存策略 FPTC (freshness-popularity-topology based caching), 考虑同一生产者在不同时刻所产生内容的新旧程度不同导致的被请求概率的不同, 结合不同内容在不同路由器的流行度以及与用户之间的距离, 得出缓存某一生产者在某一时刻所产生内容的收益值, 并依照此收益值来决定内容的缓存.

本文的主要贡献如下:

(1) 针对信息中心物联网中的缓存场景进行建模, 分析得出解决缓存优化问题的关键因素;

(2) 面向信息中心物联网场景, 提出了一种基于内容新鲜度和流行度与网络拓扑的缓存放置策略. 综合考虑不同内容的新鲜度、在不同路由器的流行度以及与用户之间的距离, 以提高缓存效率. 采用滑动窗口机制来测量不同请求的数量的变化, 确保统计的是近段时间的请求. 每个路由器根据当前每种内容的请求数及其与用户之间的跳数动态调整其收益值;

(3) 提出一种针对物联网新生成内容的收益值预测方法, 使未来收益值较高的内容更早在路由器中得到缓存, 从而提高缓存效率.

本文第 1 节介绍信息中心物联网中的缓存策略的相关工作. 第 2 节是本文建立的信息中心物联网缓存系统模型. 第 3 节介绍根据信息中心网络及物联网的相关特性所设计的缓存方案的思路和具体算法. 第 4 节是仿真实验和性能评价. 第 5 节讨论所提出的缓存策略的复杂度. 第 6 节得出结论.

1 相关工作

在信息中心网络中, 许多节点都可以存储一定量的数据. 当用户需要某些内容时, 会发送关于所需的内

容的兴趣包. 当网络中的节点收到兴趣包后, 如果该节点存有相应的请求内容后, 其将会把对应的数据作为响应, 发送数据包沿着兴趣包传播的路径原路返回, 传给用户所需的内容. 因此, 缓存策略的好坏会对缓存的性能产生较大的影响. 在信息中心网络内, 对缓存策略的研究主要可以分为两个方面: 一方面是缓存放置策略, 用来决定节点是否缓存经过的数据, 或者数据应放置在何处; 另一方面是缓存替换策略, 用来决定单个节点所缓存的数据的替换. 目前, 国内外对于信息中心网络的缓存放置策略已经有了较多的研究, 提出了 LCE (leave copy everywhere)[5], LCD (leave cache down)[18], MCD (move cache down)[18]等相关的算法, 而缓存替换策略也有 FIFO (first in first out), LRU (least recently used), LFU (least frequently used), PLRU (pre-filtering LRU)[19]等.

LCE[5]即处处缓存, 是一种早期且简单的缓存决策方案, 数据在返回请求者的途中被沿途所有的缓存节点缓存. 这种缓存决策方案虽然简单, 但是容易导致内容高度冗余, 造成较低的缓存利用率, 从而难以发挥缓存的优势. Laoutaris 等人[18]提出了一种名为 LCD 的缓存方案. 在该方案中, 内容只在缓存命中发生的节点返回请求方向的下一跳缓存, 因此能够有效降低内容的冗余度. 除了 LCD 外, 作者还提出了另一种与 LCD 类似的名为 MCD 的缓存方案. 它除了在缓存命中发生的节点返回请求方向的下一跳缓存内容外, 同时还删除了缓存命中发生的节点的缓存内容. 这两种方案均是早期针对 Web 缓存提出的, 用来降低缓存的冗余度. 在不增加缓存策略复杂性的情况下, 提高缓存多样性的最简单方法是使用概率缓存. 在这种方法的静态版本 (probabilistic caching)[20]中, 设置一个静态概率 p, 节点以概率 p 决定是否缓存到达的数据. 然而, 以静态的概率来决定是否缓存时不够灵活的, 因此诞生了基于可用信息来动态地计算每个节点甚至每个内容块的缓存概率的方案, 以使缓存行为适应网络. Psaras 等人[21]设计了一种名为 ProbCache 的动态概率缓存机制, 其根据给定内容块的生产者和请求者之间的总跳数以及到请求者的路径上剩余的跳数来计算给定内容块的缓存概率. 然而, 该方法没有考虑不同内容的不同流行度. Cai 等人[22]提出一种基于概念漂移学习的 ICN 自适应缓存策略 CDL, 在节点数据和内容数据相互感知的基础上,通过对不同环境下概念的挖掘与学习, 自适应地实现在不同概念下的缓存匹配. Liu 等人[23]提出一种在分布式缓存机制中嵌入中心式缓存决策的机制 APDR, 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. Wu 等人[24]提出了一种启发式的基于概率的缓存策略 MBP, 同时考虑了内容流行度和内容放置效益, 使每个缓存节点以一定的概率缓存通过的内容. Xu 等人[25]提出一种基于效用的缓存机制, 网络节点跟踪它们曾经缓存过的内容的效用, 路由节点协作做出缓存决策. 此外还探讨了该机制的内在权衡, 并为其在现实世界中的部署提供了指导. 这些缓存策略主要考虑的是一般的信息中心网络场景, 并不特别针对物联网场景.

目前, 有一些基于物联网节点的信息中心物联网缓存方案. Hua 等人[26]提出了一种基于雾集群的方案, 该方案同时利用了网络和终端用户设备. Song 等人[27]设计了一种智能协同缓存(SCC)方案, 有利于更好地连接设备, 并在雾计算中提供协同缓存. Din 等人[28]针对物联网内容提出了基于集群的 PUC, 该策略不断检查物联网发布服务器上的更新内容, 但这会降低物联网节点的能效. 此外, 有的方案考虑的是无线场景中的缓存. Hail 等人[29]提出了一种考虑了节点的剩余电量、节点的剩余存储容量和数据新鲜度的概率缓存方案. Jaber 和 Kacimi[30]设计了一种考虑节点中心性及其与内容源距离的缓存策略.

考虑到物联网节点计算和存储有限、能耗不易过大等因素, 本文聚焦于在路由器中缓存, 暂不考虑物联网节点缓存. 目前, 针对信息中心物联网路由器中缓存的方案大致可以分为路径上缓存和路径外缓存两类. 其中, 路径上缓存是指在消息传递的路径上进行缓存, 路径外缓存则是指在消息传递的路径外执行缓存. 文献[5,18,20,21,31]等所提出的缓存策略属于路径上缓存策略, 文献[32,33]则属于路径外缓存策略. 路径上缓存方面, Meddeb 等人[31]提出了一种名为 Consumer-cache 的缓存策略. 该策略在数据包返回路径上的每个连有消费者的缓存节点缓存数据. 该策略可能会导致缓存节点的不均衡, 即有些节点需要缓存大量内容, 而有些节点由于没有连接消费者而导致缓存空间的浪费. Chai 等人[34]提出了一种考虑了缓存节点介数的缓存方案 Betw/EgoBetw, 内容缓存在返回路径上介数最高的节点中. 然而, 该策略的复杂度高, 且在网络运行前需要全局的节点信息以及节点之间的交流开销. 因此, Pfender 等人[35]提出了一种改进的方案 ABC, 使用近似介数

来缓存数据. 这种只考虑介数的方案的潜在问题是: 增加了连接较多的节点的负载, 而对于连接较少的节点可能导致缓存空间的浪费. 本文所提方案通过在每个缓存节点中计算和缓存, 能使内容较为均衡地分配在所有节点中, 从而有效避免缓存节点负载不均以及缓存空间的浪费. Zhang 等人[36]提出了一种基于物联网数据生存期的协同缓存方案, Meddeb 等人[37]提出了一种名为 LFF 的缓存替换策略, 这两种方案均考虑了数据的新鲜度, 未考虑流行度、缓存收益、网络拓扑多种影响因素. 本文所提方案除新鲜度外, 还同时考虑了流行度、缓存收益和网络拓扑几种影响缓存效率的因素.

路径外缓存方面, Khedher 等人[32]提出了一种根据网络负载和所需服务延迟动态地将 ICN 功能分配给 ICN/IoT 节点的名为 OPA 的缓存部署算法. Nour 等人[33]提出了一种名为 NCP 的路径外缓存算法, 该算法基于物联网流量类别来选择最佳缓存位置. 然而, Meddeb 等人[31]认为: 在物联网环境中, 由于有大量的设备和大量的数据流, 指定特定的网关来处理缓存将可能导致网络内产生多个拥塞, 因此最好使用路径上缓存, 而不是使用路径外缓存将内容集中在固定的缓存节点上. 并且由于在缓存过程中需要向缓存节点发送额外的数据包, 路径外缓存会增加信令开销. 因此, 结合当前研究现状和路径上缓存的优势, 本文所提方案属于路径上缓存策略, 较路径外缓存, 能有效降低通信开销, 避免拥塞, 简化路由复杂性.

与提出新的缓存策略不同, 也有些研究对不同的缓存策略进行了对比. Zhang 等人[38]从多方面着重阐述和比较了缓存网络的优化方法, 并对缓存网络系统的理论模型研究现状加以阐述, 此外, 还分析了仍待解决的关键问题和未来的研究方向. Pfender 等人在物理实验台上使用 IoT 设备评估几种 ICN 缓存放置策略在不同拓扑下延迟和跳数减少方面的性能[39], 此外, 还在物理实验台上的 IoT 设备进行实验, 用多种性能指标评估不同的 ICN 缓存放置策略和缓存替换策略[40]. 此外也有信息中心物联网缓存安全性的研究, Guo 等人[41]研究了一种针对 NDN 网络缓存机制的攻击——虚假位置污染攻击. 通过设计一种利用兴趣多样性遍历 ISP 的 PoP 网络内路径的算法来检测和减轻这种攻击, 提出了基于概率计数和 Bloom 滤波技术的方法, 并在 NDN 路由器上实现该算法. 本文的方案和实验设计受到上述相关研究的启发.

2 信息中心物联网网内缓存模型

2.1 系统模型

由于 NDN 是 ICN 的代表性体系结构, 因此本文以 NDN 为例来阐述本文所提出的策略. 信息中心物联网场景如图 1 所示.

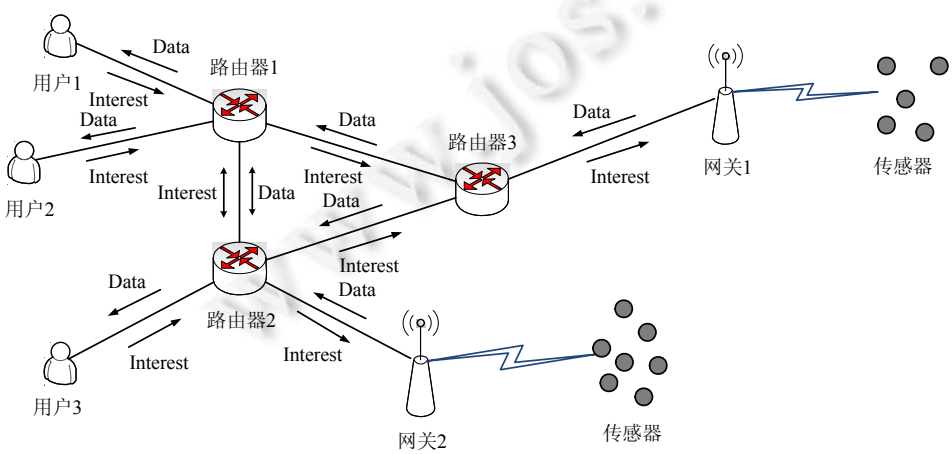


图 1 信息中心物联网场景

假设信息中心物联网由若干个物联网节点(传感器)、网关节点、路由器以及用户组成. 网络中有 Interest 包(兴趣包/请求消息)和 Data 包(数据包/数据消息)两种类型的消息. 物联网节点会产生数据并传到相连的网关

节点, 网关节点收到相应的请求时可以将与之连接的物联网节点产生的数据传回给用户. 路由器可以传递兴趣包和数据包以及缓存途经的数据. 每个路由器维护 CS, FIB 和 PIT 这 3 张表. 内容(即数据)以块为单位, 且每个内容块大小相同. 用户可以通过发送兴趣包的方式从网络中获取所需的内容, 且兴趣包会指定所需内容产生的时间. 假设内容的命名方式是层次式, 携带内容的生产者与内容产生的时间戳.

2.2 内容放置最优化模型

内容放置问题可以定义为: 在各缓存节点容量一定的前提下, 在每个时刻如何将内容块合理分配到各缓存节点, 以达到最佳目标. 相关符号定义见表 1.

表 1 信息中心物联网缓存建模中用到的符号及其含义

符号	说明
U	用户集合
P	内容提供者集合
V	路由器集合
F	内容集合
T	网络运行时间
M_p	内容提供者 p 提供的内容产生时间集合
Cv	路由器 v 的缓存容量
f_p^m	内容提供者 p 提供的在 m 时刻生成的内容
fr_p^m	内容提供者 p 提供的在 m 时刻生成的内容的新鲜度
$Q_{u,v}^{f_p^n,t}$	二分变量, t 时刻由路由器 v 满足用户 u 请求的内容 f_p^n
$Q_{u,p}^{f_p^n,t}$	二分变量, t 时刻由内容的源提供者 p 满足用户 u 请求的内容 f_p^n
$D_{v,u}$	路由器 v 与用户 u 之间的延迟
$D_{p,u}$	内容提供者 p 与用户 u 之间的延迟
$y_{f_p^m,v}^t$	二分变量, t 时刻路由器 v 是否缓存有内容 f_p^m
$y_{f_p^m,p}$	二分变量, p 是否是内容 f_p^m 的源提供者

网络内缓存的最主要目的是降低整个网络的流量以及用户的访问延迟, 这样对互联网服务提供商和用户都是有利的. 而要实现这样的目标, 可以通过减少对内容源提供者的访问次数并且使内容尽可能靠近用户来实现. 因此, 针对 ICN 缓存内容放置问题, 定义两个最终目标, 分别如下所示.

(1) 目标 1: 尽可能降低作为内容提供者的物联网节点负载(降低网络流量):

$$\mathrm{Min}\sum_{\substack{u\in U\\ p\in P}}\sum_{\substack{f_p^m\in F\\ m\in M_p}}\sum_{t,n\in T} y_{f_p^m,p}\cdot Q_{u,p}^{f_p^n,t} \tag{1a}$$

$$f_p^m = f_p^n \ \text{iff}\ m \leqslant n < m + fr_p^m \tag{1b}$$

其中, $Q_{u,p}^{f_p^n,t}$ 表示 t 时刻是否由内容的源提供者 p 满足用户 u 的请求. 因此, 使其最小化即可尽可能降低内容提供者的负载. 公式(1b)表明, 内容与用户的请求匹配的条件是: 用户所要求的内容产生时间在内容的实际产生与过期之间.

(2) 目标 2: 尽可能降低内容访问延迟:

$$\mathrm{Min}\sum_{\substack{u\in U\\ v\in V}}\sum_{\substack{f_p^m\in F\\ m\in M_p}}\sum_{t,n\in T} y_{f_p^m,v}^t\cdot Q_{u,v}^{f_p^n,t}\cdot D_{v,u} \tag{2a}$$

$$f_p^m = f_p^n \ \text{iff}\ m \leqslant n < m + fr_p^m \tag{2b}$$

其中, $y_{f_p^m,v}^t$ 表示 t 时刻路由器 v 是否缓存有内容 f_p^m, $Q_{u,v}^{f_p^n,t}$ 表示 t 时刻是否由路由器 v 满足用户 u 请求的内容 f_p^n, $D_{v,u}$ 表示路由器 v 与用户 u 之间的延迟. 因此, 三者的乘积表示由路由器满足用户请求的延迟, 使其最小化即可, 尽可能降低内容检索延迟. 公式(2b)表明内容与用户请求匹配的条件.

由以上两个目标, 可构造如下优化模型:

$$\text{Min} \sum_{\substack{u\in U\\ p\in P, v\in V}} \sum_{\substack{f_p^m\in F\\ m\in M_p}} \sum_{t,n\in T} y_{f_p^m,p}\cdot Q_{u,p}^{f_p^n,t} + y_{f_p^m,v}^t \cdot Q_{u,v}^{f_p^n,t}\cdot \frac{D_{v,u}}{\sum_{p\in P} y_{f_p^m,p} D_{p,u}} \tag{3}$$

$$\text{s.t. } y_{f_p^m,v}^t \in \{0,1\},\ \forall v\in V,\ f_p^m \in F,\ t\in T,\ m\in M_p \tag{4}$$

$$y_{f_p^m,p} \in \{0,1\},\ \forall p\in P,\ f_p^m \in F,\ m\in M_p \tag{5}$$

$$Q_{u,v}^{f_p^n,t} \in \{0,1\},\ \forall u\in U,\ v\in V,\ f_p^n \in F,\ t\in T,\ n\in T \tag{6}$$

$$Q_{u,p}^{f_p^n,t} \in \{0,1\},\ \forall u\in U,\ p\in P,\ f_p^n \in F,\ t\in T,\ n\in T \tag{7}$$

$$f_p^m = f_p^n \text{ iff } m\leqslant n < m + fr_p^m \tag{8}$$

$$\sum_{v\in V} Q_{u,v}^{f_p^n,t} + Q_{u,p}^{f_p^n,t} = 1,\ \forall u\in U,\ p\in P,\ f_p^n \in F,\ t\in T,\ n\in T \tag{9}$$

$$\sum_{f_p^m\in F} y_{f_p^m,v}^t \leqslant C_v,\ \forall v\in V,\ t\in T,\ m\in M_p \tag{10}$$

$$\sum_{f_p^m\in F} y_{f_p^m,p} = 1,\ \forall p\in P,\ m\in M_p \tag{11}$$

其中, 最终目标与前面分开目标的不同之处在于不是前两个目标的简单相加, 而是将目标 2 化为无量纲后再进行相加. 最终目标中的分式表明了用户 u 到满足用户 u 请求的路由器之间的延迟和用户 u 到内容源提供者之间的延迟的比值, 比值越小, 说明满足用户 u 的请求路由器越接近用户 u, 因此可间接表示目标 2. 模型的目标是求出每个时刻的 $y_{f_p^m,v}^t$, $Q_{u,v}^{f_p^n,t}$ 和 $Q_{u,p}^{f_p^n,t}$, 使公式(3)的值最小化. 公式(4)~公式(7)表明这 4 个变量为二分变量; 公式(8)表示内容与用户的请求匹配的条件是, 用户所要求的内容产生时间在内容的实际产生与过期之间; 公式(9)表示 t 时刻用户 u 的一个请求只能由一个节点来满足; 公式(10)表示任意时刻路由器缓存的内容均不能超过其缓存大小; 公式(11)表示每个内容有且仅有一个源提供者.

2.3 问题分析

由于目标函数与约束条件均为线性, 且自变量 $y_{f_p^m,v}^t$, $Q_{u,v}^{f_p^n,t}$ 和 $Q_{u,p}^{f_p^n,t}$ 的取值均为整数, 因此该问题是整数线性规划问题(integer linear programming, ILP). 由于该问题任意一个可能的解都可以在多项式时间内判断其是否是可行解, 因此该问题是一个 NP 问题. 不难发现, 该 ILP 问题可归约为 3-CNF-SAT 问题[42], 因此其是 NP 完全问题, 要直接求解是困难的.

同时, 如果直接求解该问题, 得到的是一种全局集中式的内容放置方案, 计算时需要全局的各种信息, 还需要集中控制节点来实现内容分配, 难以在分布式的缓存系统中实现. 此外, 由于网络会持续运行, 想要在一开始就求得总时间内的最优解是不现实的. 因此, 本文提出一种分布式缓存内容放置算法, 作为一种实际可行的方案, 所有路由器基于局部信息独立进行缓存决策.

观察模型可以发现, $Q_{u,v}^{f_p^n,t}$ 和 $Q_{u,p}^{f_p^n,t}$ 是其中关键的参数, 其关系到缓存的内容是否有用. 如果分别对两个参数在一段时间里求和, 可以得出某个内容在某个节点处被满足的次数, 其类似于该内容在某个节点处的流行度. 如果可以获得一段时间内每个内容在每个节点的流行度, 就可以更好地决定该段时间内每个节点需要缓存什么内容以达到最佳目标. 因此, 内容的流行度是求解问题的一个关键因素. 观察 $Q_{u,v}^{f_p^n,t}$ 和 $Q_{u,p}^{f_p^n,t}$ 可以发现, f_p^n 是二者共有的一个元素, 其表明内容的提供者及内容的产生时间. 由前面对物联网的研究可知: 在物联网环境中, 同一个生产者在不同时刻所产生的数据在同一时刻被请求的概率是不一样的, 而且内容与用户的请求匹配需要满足公式(8)的条件. 因此, 内容的新鲜度与新旧程度对于求解的问题来说也是关键的因素. 此外, 为了尽可能降低内容访问延迟, 需使 $D_{v,u}$ 尽可能地小, 即内容缓存的地方与用户之间的跳数需要尽可能小. 基

于以上关键因素, 本文提出一种基于内容新鲜度、新旧程度和流行度以及网络拓扑的缓存内容放置策略, 缓存节点可以采用分布式的内容缓存策略, 以优化缓存系统的整体性能.

3 信息中心物联网缓存放置策略

3.1 策略概述

NDN 路由器各自按照 FPTC 策略对路过的数据包内容进行缓存. 每个 NDN 路由器记录最近一段时间内收到的请求, 并计算缓存每种请求对应的数据包的收益值; 当数据包到达时查找其对应的收益值排名, 若排名在 NDN 路由器的缓存容量以内, 则将其缓存. 对于兴趣包, 请求消息中有对内容产生的时间的要求信息, 并新增一个名为 ITH (interest trace hop)的字段, 以记录兴趣包自发出至当前所到达处的跳数, 跳数越小则在该路由器缓存对应内容的收益越高; 对于数据包, 内容名字中包含内容产生的时间与新鲜度信息, 其中, 新鲜度表示内容生产者生成内容与该内容失效(即过期)之间的时间长度. 若请求消息中的时间在内容产生时间与内容过期时间之间, 则表示内容满足请求消息. 每个 NDN 路由器在原有 CS, FIB 和 PIT 这 3 张表的基础上新增一个请求队列(*request_queue*)和一张统计表(*statistical_table*). 其中, 请求队列用来记录最近一段时间内, 路由器接收到的请求消息, 队列中的每个元素为请求消息所请求内容的名称. 由于请求队列满时会发生队首元素的剔除与队尾元素的插入, 因此其类似一个请求序列的滑动窗口. 统计表根据请求队列中的元素, 统计最近一段时间内的每种内容的请求数及相应的平均跳数, 并根据本文第 3.2.2 节与第 3.2.3 节所提出的计算方法计算出每种内容的流行度与收益值. 若路由器收到一个统计表中没有记录的新请求时, 会检查统计表中与新请求属于同一生产者的其他条目, 并根据本文第 3.2.3 节的方法预测其收益值. 请求队列的设置不仅保证路由器所统计的是近期请求, 同时也避免了统计表的条目无限增多. 在路由器接收到数据包时, 会检查统计表, 如果内容的收益值在表中的排名小于路由器的最大存储容量大小, 该内容将被缓存. 兴趣包与数据包的结构如图 2 所示. 兴趣包中含有请求的内容名、请求的内容的时间戳和 ITH 这 3 个字段; 数据包含有内容名、内容产生的时间戳、内容的新鲜度这 3 个字段以及内容. 统计表的结构见表 2, 表中的每个条目包含 6 个字段: 内容名、生产者、时间戳、请求数、跳数和收益值. 内容名即兴趣包所请求的内容的名称; 生产者表示所请求的内容的源生产者, 时间戳表示生产者产生相应内容的时间, 这两列的设置便于流行度计算时对条目的分类和排序; 请求数表示请求队列中对应的请求消息的数量; 跳数表示请求队列中对应的请求消息的平均跳数; 收益值即通过计算所得的请求的内容在该路由器中的收益值.

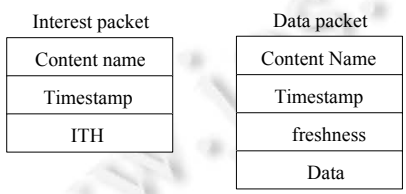


图 2 兴趣包与数据包结构

表 2 统计表结构

内容名	生产者	时间戳	请求数	跳数	收益值
…/P_1/$timestamp_1$	P_1	$timestamp_1$	$Req_{1,1}$	$Hop_{1,1}$	$Reward_{1,1}$
…/P_1/$timestamp_2$	P_1	$timestamp_2$	$Req_{1,2}$	$Hop_{1,2}$	$Reward_{1,2}$
…/P_2/$timestamp_1$	P_2	$timestamp_1$	$Req_{2,1}$	$Hop_{2,1}$	$Reward_{2,1}$
…	…	…	…	…	…

3.2 具体算法

3.2.1 内容的新鲜度

在物联网环境中, 物联网每隔一段时间会产生一个新的内容, 每个内容均有新鲜度, 仅能代表某段时间的值, 过了这段时间即意味着有新的内容产生, 旧的内容过期, 不再新鲜. 例如, 某温度传感器在今日 1:00 产

生一个新鲜度为 10 分钟的值, 则该值只能代表今日 1:00~1:10 期间的温度. 用户对内容的时间也会有要求, 例如查询最新的温度或者昨日 12:00 的温度. 因此, 要使内容与请求相匹配, 需要考虑内容的新鲜度与用户所请求的内容的时间. 为了更好地统计内容的收益值以及防止统计表中的条目过多, 兴趣包中的时间戳满足如下规则: 若用户之前未请求过对应的生产者产生的内容, 则兴趣包中的时间戳为用户请求的内容的时间戳; 否则, 用户会记录有对应生产者产生的内容的新鲜度及历史产生时间, 从而可以获得与用户需求相匹配的内容的时间戳, 此时, 兴趣包中的时间戳满足公式:

$$Interest.Timestamp \leqslant User.Timestamp < Interest.Timestamp + Data.freshness \tag{12}$$

其中, *User.Timestamp* 表示用户请求时刻为 *Timestamp* 的内容, *Interest.Timestamp* 为用户发出的兴趣包中的时间戳, *Data.freshness* 为对应生产者产生的内容的新鲜度.

3.2.2 内容流行度与跳数的计算

每个路由器所接收到的请求消息及其数量都不相同, 因此, 通过计算每种请求的流行度判断接收到的请求消息在自己节点的流行程度, 并以此来决定是否缓存接收到的数据是有意义的. 在每个路由器中, 每种内容的流行度即对该内容的请求数占所有请求的比例, 因此, 对每一个内容生产者 P_i 在 t_j 时刻所产生内容的流行度计算公式为

$$Popularity(P_i,t_j)=\frac{R_{P_i,t_j}}{\sum_{m=1}^{M}\sum_{n=1}^{N_{P_m}}R_{P_m,t_n}} \tag{13}$$

其中, R_{P_i,t_j} 表示在统计表中, 对内容生产者 P_i 在 t_j 时刻所产生内容的请求数; M 表示在统计表中, 共有 M 个不同的生产者; N_{P_m} 表示在统计表中, 有对生产者 P_m 在 N_{P_m} 个不同时刻产生内容的请求.

如前所述, 物联网中一个重要特征是: 每个物联网节点每隔一段时间会产生一个新的内容, 对于不同时间产生内容的值是不同的, 而且物联网环境中的用户更趋向于请求最新的信息. 因此, 内容的新旧会对收益值产生影响. 在统计表中, 将同一个内容生产者在不同时刻所产生的内容按照新旧排序, 最新的内容被请求的概率最大, 最旧的内容被请求的概率最小. 假设以时间戳区分同一个内容生产者在不同时刻所产生的内容, 时间戳越大, 则表示内容越新, 该时间戳作为内容名称中的一部分. 对同一个内容生产者 P_i 在不同时刻所产生的内容, 有:

$$\sum_{k\in N_{P_i}} prob(P_i,t_k)=1 \tag{14}$$

$$prob(P_i,t_m)>prob(P_i,t_n),\ \forall t_m>t_n \tag{15}$$

其中, N_{P_i} 表示在统计表中, 有对生产者 P_i 在 N_{P_i} 个不同时刻产生内容的请求; $prob(P_i,t_k)$表示时间戳为 t_k 的内容被请求的概率. 公式(14)表示: 在统计表中, 请求同一个生产者 P_i 在不同时刻产生的内容概率和为 1. 公式(15)表示: 对于同一个生产者 P_i 在不同时刻所产生的内容, 新内容被请求的概率比旧内容被请求的概率大.

为了降低内容的访问延迟, 内容应尽可能缓存在离请求的用户近的地方. 因此, 本文方案将网络拓扑纳入考虑来计算缓存内容的整体收益. 在兴趣包中加入一个名为 ITH 的字段, 以记录兴趣包自发出至当前路由器的跳数. 路由器中的统计表由跳数字段记录请求到达该路由器的平均跳数, 计算方法为

$$hop(P_i,t_j)=\begin{cases} ITH(P_i,t_j), & (P_i,t_j)\text{ is a new entry in router} \\ \dfrac{hop(P_i,t_j)+ITH(P_i,t_j)}{2}, & (P_i,t_j)\text{ is an existed entry in router} \end{cases} \tag{16}$$

其中, $ITH(P_i,t_j)$为新到达的兴趣包中的 ITH 字段值, $hop(P_i,t_j)$为统计表中条目(P_i,t_j)的 hop 字段值. 此处用统计表中的原 hop 值与 ITH 值做平均得到一个总平均值的近似值.

3.2.3 缓存内容收益的计算与预测

统计表中每个条目的总收益根据上一小节计算的几个值获得, 计算公式为

$$Reward(P_i,t_j)=\frac{Popularity(P_i,t_j)\cdot prob(P_i,t_j)}{hop(P_i,t_j)} \tag{17}$$

其中, $Popularity(P_i,t_j)$由公式(13)计算获得, $prob(P_i,t_j)$根据公式(14)与公式(15)使用特定方式计算获得, $hop(P_i,t_j)$由公式(16)获得. 由公式(17)可知: 内容越新、流行度越大、路由器与请求相应内容的用户之间跳数越小, 缓存内容的收益就越高.

在物联网中, 生产者往往每隔一段时间便会产生新的数据, 因此, 路由器常常会收到对新内容的请求. 当路由器收到统计表中没有记录的请求时, 仅通过上述计算方法可能会导致新请求的流行度和跳数与后续稳定时有较大的偏差, 导致新请求的收益值与后续稳定时差距较大. 为了使统计表中的新条目的总收益能更早得到较为准确的值, 本方案对统计表中的新条目的收益值进行预测. 在本文所针对的场景及所提出的方案中, 路由器所维护的请求队列和统计表记录近期请求, 因此历史数据量比较少; 此外, 路由器的计算能力有限, 不适合使用过于复杂的算法. 由于灰色系统预测模型[43]具有无需大量数据样本、短期预测效果好和运算过程简单等特点, 适用于本文场景及方案, 因此, 本方案采用灰色系统预测模型进行预测.

预测统计表中新条目收益值的主要思路如下: 首先, 在统计表中找出与新条目属于同一生产者的条目, 得到比新条目更早产生的数据的收益值, 得到一个收益值序列. 假设得到的收益值序列为 $x^{(0)}=(x^{(0)}(1), x^{(0)}(2),\ldots,x^{(0)}(n))$, 待预测的收益值为 $x^{(0)}(n+1)$. 接下来, 使用灰色系统预测模型, 通过收益值序列得到新条目的流行度的预测值, 作为新条目的收益值.

本方案灰色模型为 $GM(1,1)$模型. 首先对收益值序列进行级比检验, 判断使用 $GM(1,1)$建模的可行性. 序列的级比定义为

$$\sigma^{(0)}(k)=\frac{x^{(0)}(k-1)}{x^{(0)}(k)},\ k=2,3,...,n \tag{18}$$

如果满足 $\sigma^{(0)}(k)\in\left(\mathrm{e}^{-\frac{2}{n+1}},\mathrm{e}^{\frac{2}{n+1}}\right)$, 则认为收益值序列 $x^{(0)}$可用 $GM(1,1)$建模. $GM(1,1)$建模过程如下.

(1) 对收益值序列 $x^{(0)}$进行累加, 从而弱化序列的波动性和随机性, 得到新的序列 $x^{(1)}=(x^{(1)}(1),x^{(1)}(2),\ldots,x^{(1)}(n))$. 累加的方法为

$$x^{(1)}(k)=\sum_{i=1}^{k}x^{(0)}(i),\ k=1,2,...,n \tag{19}$$

(2) 生成 $x^{(1)}$的邻均值等权序列 $z^{(1)}=(z^{(1)}(2),z^{(1)}(3),\ldots,z^{(1)}(n))$, 其中,

$$z^{(1)}(k)=0.5x^{(1)}(k-1)+0.5x^{(1)}(k),\ k=2,3,\ldots,n \tag{20}$$

(3) 用最小二乘法估计灰参数 a 和 u:

$$\hat{U}=\begin{bmatrix}\hat{a}\\ \hat{u}\end{bmatrix}=(B^{\mathrm{T}}B)^{-1}B^{\mathrm{T}}Y_n \tag{21}$$

其中, a 为发展系数, u 为灰色作用量. 矩阵 B 与常数项向量 Y_n 由步骤(1)和步骤(2)中生成的序列 $x^{(1)}$和 $z^{(1)}$生成: $Y_n=(x^{(0)}(2),x^{(0)}(3),\ldots,x^{(0)}(n))$, $B=\begin{bmatrix}-z^{(1)}(2) & -z^{(1)}(3) & \ldots & -z^{(1)}(n)\\ 1 & 1 & \ldots & 1\end{bmatrix}^{\mathrm{T}}$;

(4) 通过 a 与 u 的估计值求解, 得:

$$\hat{x}^{(1)}(k+1)=\left[x^{(1)}(1)-\frac{\hat{u}}{\hat{a}}\right]\mathrm{e}^{-\hat{a}t}+\frac{\hat{u}}{\hat{a}},\ k=1,2,...,n \tag{22}$$

(5) 将步骤(4)的结果进行累减还原:

$$\hat{x}^{(0)}(k+1)=\hat{x}^{(1)}(k+1)-\hat{x}^{(1)}(k),\ k=1,2,...,n \tag{23}$$

最后得到的序列为 $\hat{x}^{(0)}=(\hat{x}^{(0)}(1),\hat{x}^{(0)}(2),...,\hat{x}^{(0)}(n),\hat{x}^{(0)}(n+1))$, 其中, $\hat{x}^{(0)}(n+1)$ 即为新条目的预测收益值.

3.2.4 处理兴趣包

路由器接收到兴趣包时, 将其所请求的内容名放入到请求队列中, 如果请求队列已满, 则先取出队首元素并将其删除. 此后, 在统计表中, 根据请求队列的变化重新对每个请求内容的收益值进行计算, 得到新的收益值. 此外, 若在请求到达时统计表中没有记录该请求, 即该请求是统计表中的新条目, 则使用该请求的预测

收益值作为其收益值. 具体描述如算法 1 所示.

- 第 1–7 行是兴趣包到达时对兴趣包中的 ITH 字段的设置以及消息的转发;
- 第 8–9 行是兴趣包到达路由器时请求队列和统计表中请求数和跳数的修改;
- 第 10–15 行针对统计表产生新条目时对新条目的收益值进行预测;
- 第 16 行是在统计表变动后对收益值的重新计算.

算法 1. 兴趣包处理算法.

```
Input: 兴趣包 I; 请求队列 request_queue; 统计表 statistical_table;
1  I.ITH=I.ITH+1;
2  IF canSatisfy(I) THEN   /*若 CS 能满足兴趣包, 则设置数据的两个新增字段, 并传回对应的数据*/
3     reply(Data);
4  END IF
5  ELSE   /*若 CS 不能满足兴趣包, 则转发兴趣包*/
6     forward(I);
7  END ELSE
8  request_queue.insert(I);   /*到达的请求插入到请求队列*/
9  statistical_table.update;   /*统计表根据请求队列中元素的变化, 修改对应条目的请求数和跳数*/
10 IF isNewEntry(I) THEN   /*如果是新条目, 则尝试进行预测*/
11       data=getHistoryReward(I);   /*获得统计表中与 I 为同一生产者的历史数据*/
12       IF data.size(·)>1 THEN   /*如果有两个或以上的历史数据, 则进行预测*/
13          statistical_table[I.name].reward=GreyPrediction(data);
14       END IF
15 END IF
16 calculateReward(statistical_table);   /*重新计算条目的收益值, 若是新条目, 则用预测值*/
```

3.2.5 处理数据包

当数据包到达路由器时, 路由器将根据统计表中的具体收益值排名决定是否缓存该数据包. 收益值越高, 排名越高. 当在统计表中找到对应的条目并且其收益值在表中的排名小于等于路由器的 CS 大小时, 该数据将会被缓存在 CS 中. 具体描述如算法 2 所示.

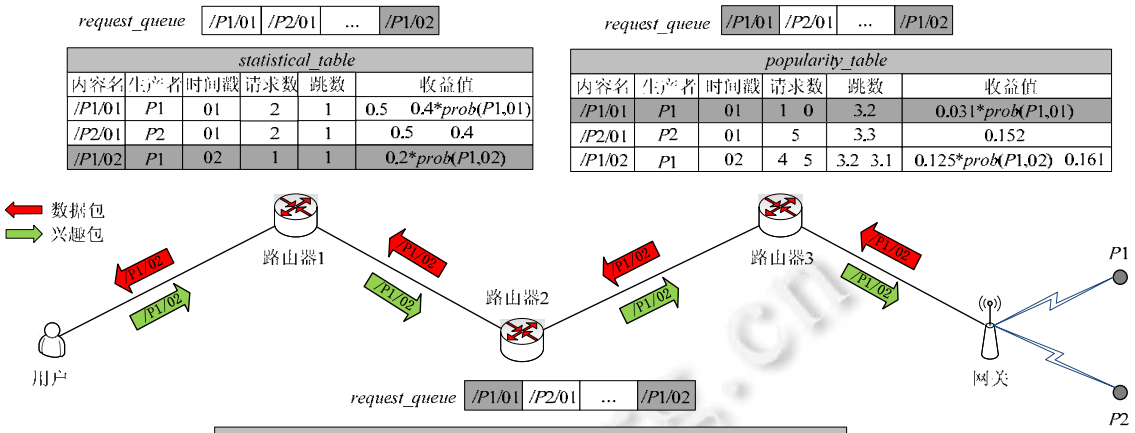
算法 2. 数据包处理算法.

```
Input: 数据 D; 统计表 statistical_table; 内容存储 CS;
1  IF D.name in statistical_table THEN   /*当在统计表中找到 D 对应的条目时, 决定是否缓存*/
2     IF rank(statistical_table[D.name])≤CS.size THEN   /*若排名小于等于 CS 大小, 则缓存*/
3        cache(D);
4     END IF
5  END IF
6  forward(D);   /*转发数据包*/
```

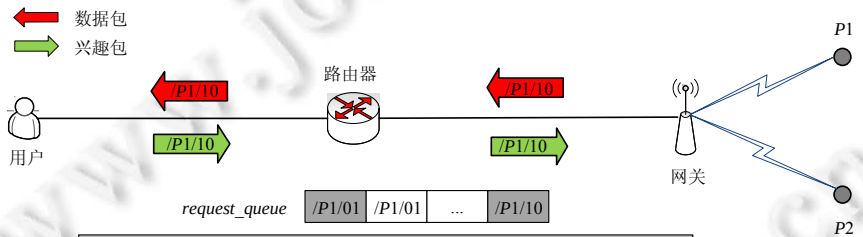
3.3 FPTC工作过程示例

本小节将给出具体的示例, 来说明所提出的缓存策略 FPTC 的工作过程.

首先介绍请求队列和统计表的维护过程. 当统计表中原有新到达的请求所属条目, 或者缺少与新条目同属一个生产者的条目时, 就不会用到灰色预测. 图 3(a)中, 3 个路由器在兴趣包"/P1/02"到达时均未使用到预测, 路由器 2 和路由器 3 属于前一种情况, 路由器 1 属于后一种情况.



(a) 未用到灰色预测情况示例



statistical_table

内容名	生产者	时间戳	请求数	跳数	收益值
/P1/01	P1	01	3 2	2.2	popularity(P1,01)*prob(P1,01)/2.2 popularity(P1,01)*prob(P1,01)/2.2
/P2/01	P2	01	3	2.1	popularity(P2,01)*prob(P2,01)/2.1
/P1/02	P1	02	4	2.3	popularity(P1,02)*prob(P1,02)/2.3
/P1/03	P1	03	5	1.9	popularity(P1,03)*prob(P1,03)/1.9
...	...	...	...	...	...
/P1/10	P1	10	1	1	Prediction(P1,10)

(b) 使用灰色预测情况示例

图 3 FPTC 工作过程示例

如图 3(a)所示: 网关节点与传感器 *P*1 和 *P*2 连接, 可以为用户提供 *P*1 和 *P*2 产生的数据, 每个路由器最多能缓存 2 个内容. 用户发出一个兴趣包, 请求生产者 *P*1 时间为 02 的内容, 名称假设为"/*P*1/02". 由于所有路由器的缓存中均未有对应的数据, 因此该兴趣包经由路由器传到了网关节点. 在兴趣包到达路由器 1 时, 由于请求队列未满, 请求的名称直接插入队列中(深色元素), 由于队列中原来没有名称为"/*P*1/02"的元素, 统计表中原来也没有对应的条目, 因此统计表中新增一个条目(深色条目), 并重新计算每种内容的收益值: "/*P*1/01"的收益值由 $2/(2+2)\cdot 1\div 1=0.5$ 变为 $2/(2+2+1)\cdot prob(P1,01)\div 1=0.4\cdot prob(P1,01)$, "/P2/01"的收益值由 $2/(2+2)\cdot 1\div 1=0.5$ 变为 $2/(2+2+1)\cdot 1\div 1=0.4$, "/*P*1/02"的收益值为 $2/(2+2+1)\cdot prob(P1,02)\div 1=0.2\cdot prob(P1,02)$. 当兴趣包到达路由器 2 时, 由于请求队列已满, 队首元素为"/*P*1/01", 因此将队首元素(深色元素)弹出队列, 再将"/*P*1/02"插入队尾(深色元素). 相应地, 统计表中"/*P*1/01"和"/*P*1/02"的请求数量发生变化, 并重新计算每种内容的收益值: "/*P*1/01"的收益值由 $0.192\cdot prob(P1,01)$ 变为 $0.154\cdot prob(P1,01)$, "/*P*2/01"的收益值仍为 0.107,

“/*P*1/02”的收益值由 0.08·*prob*(*P*1,02)变为 0.133·*prob*(*P*1,02). 当兴趣包到达路由器 3 时, 请求队列的情况与路由器 2 一样, 而在统计表中, 由于“/*P*1/01”的请求数量原来为 1, 在减 1 之后会变为 0, 因此将该条目删除(深色条目), “/*P*1/02”的请求数量加 1, 然后重新计算收益值: “/*P*2/01”的收益值仍为 0.152, “/*P*1/02”的收益值由 0.125·*prob*(*P*1,02)变为 0.161. 网关节点在收到兴趣包后, 返回一个对应的数据包. 当数据包到达路由器时, 路由器对其统计表中的条目按照收益值由高到低进行排名, 若数据包对应的内容收益值排名为 1 或 2 时, 路由器会缓存数据.

接下来给出一个用到灰色预测的示例. 当统计表中与新条目同属一个生产者的条目足够多时, 就会使用灰色预测对新条目的收益值进行预测. 因此, 当网络持续运行时, 很多时候会用到预测. 如图 3(b)所示, 网关节点与传感器 *P*1 和 *P*2 连接, 可以为用户提供 *P*1 和 *P*2 产生的数据. 路由器中的统计表中原有条目为 *P*1 在时间 01 到 09 产生的内容名(即“/*P*1/01”至“/*P*1/09”), 以及 *P*2 在时间 01 到 05 产生的内容名(即“/*P*2/01”至“/*P*1/05”). 用户发出一个兴趣包, 请求生产者 *P*1 时间为 10 的内容, 名称假设为“/*P*1/10”. 当兴趣包到达路由器时, 由于请求队列已满, 队首元素为“/*P*1/01”, 因此, 将队首元素(深色元素)弹出队列, 再将“/*P*1/10”插入队尾(深色元素). 相应地, 统计表中“/*P*1/01”的请求数量减 1. 由于队列中原来没有名称为“/*P*1/10”的元素, 统计表中原来也没有对应的条目, 因此统计表中新增一个条目(深色条目). 由于这是一个新条目, 因此尝试预测其收益值. 首先, 从统计表中找出与新条目属于同一生产者并且时间戳小于新条目的时间戳的所有条目, 即“/*P*1/01”至“/*P*1/09”共 9 个条目, 将这 9 个条目按时间从小到大的顺序排列, 取出 9 个条目的收益值得到一个收益值序列 $x^{(0)}=(x^{(0)}(1),x^{(0)}(2),\ldots,x^{(0)}(9))$, 设待预测的新条目收益值为 $x^{(0)}(10)$. 接下来计算序列的级比 $\sigma^{(0)}(k)=x^{(0)}(k-1)/x^{(0)}(k)$, $k=2,3,\ldots,9$ 是否在区间 $\left(\mathrm{e}^{-\frac{2}{9+1}},\mathrm{e}^{\frac{2}{9+1}}\right)$ 内, 如果满足, 则按照公式(19)~公式(23)计算得到 $\hat{x}^{(0)}(10)$ 即为新条目的预测收益值, 假设为 *Prediction*(*P*1,10). 接下来重新计算所有条目的收益值, 填入统计表, 新条目的收益值用 *Prediction*(*P*1,10)的数值填入.

4 仿真实验

4.1 实验设置

为了对所提出缓存放置策略 FPTC 的性能进行评价, 本文在 ndnSIM[44]上进行了相关仿真实验. ndnSIM 平台基于 NS-3[45], 将 NDN 的功能设计成一个独立的协议栈, 可以方便地配置到节点之上, 实现 NDN 网络功能部署. FPTC 将与 5 种现有的其他缓存决策策略进行性能对比.

(1) LCE[5]: 缓存决策最简单的方法, 让每个节点缓存每个传入内容;

(2) LCD[18]: 内容只在缓存命中发生的节点返回请求方向的下一跳缓存;

(3) Probabilistic Caching[20]: 每个缓存节点以固定概率缓存传入的内容, 本实验中设置此概率为 0.5 (p=0.5);

(4) ProbCache[21]: 每个缓存节点缓存传入内容的概率是动态的, 根据缓存节点与发出请求的用户之间的距离来确定缓存概率, 越接近发出请求的用户的缓存节点, 缓存概率越高;

(5) ABC[35]: 将内容缓存在返回路径中近似中心度最高的节点.

对于所有的缓存决策策略, 均使用 LRU 作为缓存替换策略, 使用最短路径路由(SPR)作为转发策略.

实验采用图 4 所示的网络拓扑结构. 在两种拓扑结构中, 各有 9 个不同的网关节点、15 个不同的用户, 在拓扑 1 中有 20 个路由器, 在拓扑 2 中有 8 个路由器. 每个网关节点连有 10 个作为内容生产者的物联网节点, 因此, 共有 90 个不同的物联网节点. 作为内容提供者的网关节点与物联网节点相连, 可以提供与之相连的物联网节点所产生的数据给用户. 用户发送的请求最远在与对应的内容生产者(即物联网节点)相连的内容提供者(即网关节点)处可以获得数据. 实验设置相邻节点之间的带宽为 1 Mb/s, 延迟为 2 ms.

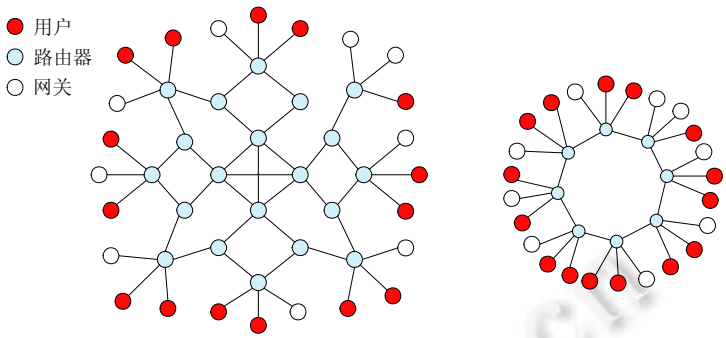


图 4 实验拓扑图

实验设置不同的内容生产者被请求概率的概率相同. 每个用户随机请求 12 个不同生产者的内容. 每个内容生产者每隔 1 s 会产生一个新的内容, 即每个内容的新鲜度为 1 s. 实验设置同一个内容生产者在不同时间产生的内容被请求的概率服从如下规则:

$$prob(P_i,t_r)=\begin{cases}[0.8,1.0], & t_r=t_{latest}\\ Zipf:\alpha=0.8,C=1, & \text{else}\end{cases} \tag{24}$$

其中, r 表示生产者 P_i 产生的第 r 个内容, t_r 表示内容产生时间, r 越大, 即内容越新; t_{latest} 表示最新内容的产生时间. 公式(24)表示: 用户对同一个内容生产者的请求中, 最新内容被请求的概率最大, 为[0.8,1.0]; 对其他较旧的内容的概率由剩下的[0,0.2]的概率按照参数α=0.8, C=1 的 Zipf 分布[46]进行分配.

在实验中, 通过改变每个节点最大可以缓存的内容块数(*CS* 的大小)、请求最新内容的概率、每个用户的请求速率、网络拓扑以及所提出的 FPTC 缓存策略中请求队列的长度来观察性能变化的趋势. 具体实验参数变化设置见表 3. 每次实验取仿真开始 2 s 之后 30 s 的数据作为实际处理的数据时间, 每种情况均做 10 次实验, 结果取平均值. 在本实验中, 90 个内容生产者每隔 1 秒都会产生新内容, 因此整个网络开始运行时的内容总数为 90, 在运行的 30 s 内一共会产生 90×30=2700 个不同的内容. 在文献[47]中, Rossi 和 Rossini 证明了缓存的大小 C 与目录大小(网络中的内容总数)F 的比值为$[10^{-5},10^{-1}]$. 在本实验中, 设置 C 与 F 的比值范围约为 10^{-3} 到 10^{-2} 之间. 为了体现 ICN 的缓存所带来的好处, 增加一个 CS 大小为 0(即没有缓存)的情况. 在本实验中, 设置 CS 的大小取值为[0,10], 每个用户的请求速率取值为 5 个/s 至 20 个/s. FPTC 策略中, 每个路由器的请求队列长度(最大容纳的请求条数)为 50–850. 此外, 实验设置内容生产者产生的数据大小均为 1 KB.

表 3 实验参数

参数	默认值	变化范围
内容数	–	[90,2700]
缓存容量	6	[0,10]
访问模式	最新内容请求概率: 0.9; 其他内容请求概率: *Zipf*: α=0.8	最新内容请求概率: [0.8,1.0]
请求速率	10 个/s	[5,20]
请求队列长度	250	[50,850]
实验拓扑	拓扑 1	(拓扑 1,拓扑 2)

4.2 性能指标

为了反映不同缓存决策策略的实际效果、评价不同缓存方案的性能, 本文定义了以下 4 个性能评价指标.

(1) 平均访问延迟(average access delay, AAD)

平均访问延迟表示从用户发出请求消息(即兴趣包)到收到对应的内容(即数据包)所经历的平均延迟. 本指标反映缓存服务的响应速度. 本文考虑用户至网关节点之间的延迟, 不考虑物联网节点到网关节点之间的延迟. 平均访问延迟越低, 缓存效率越高. 平均访问延迟的计算方法为

$$AAD=\frac{\sum_{i=1}^{q}d_i}{q} \tag{25}$$

其中, d_i 表示用户从发出请求 i 到收到请求 i 对应的内容所经历的延迟, q 表示请求的总数.

(2) 平均跳数(average hop count, AHC)

平均跳数表示缓存命中发生的节点与用户发出请求所在的节点之间的平均跳数距离. 与上一个指标相同, 只考虑用户至网关节点之间的跳数. 平均跳数越低, 表示缓存命中发生的节点越接近用户发出请求所在的节点. 由于延迟与跳数有着紧密的相关性, 因此该指标可作为对平均访问延迟的补充. 平均跳数的计算方法为

$$AHC=\frac{\sum_{i=1}^{q}hop_i}{q} \tag{26}$$

其中, hop_i 表示请求 i 对应的内容命中处(即开始发送数据包处)到用户所经历的跳数, q 表示请求的总数.

(3) 缓存服务率(cache service ratio, CSR)

缓存服务率反映了用户的请求由路由器的缓存而不是内容提供者响应的比率. 缓存服务率越高, 缓存系统的效率就越高. 缓存服务率的计算方法为

$$CSR=\frac{\sum_{i=1}^{r}hit_i}{\sum_{i=1}^{r}hit_i+\sum_{j=1}^{P}hit_j} \tag{27}$$

其中, r 表示路由器的数量, hit_i 表示在路由器 i 缓存命中的次数. 因此, 分子表示在中间路由器请求命中的次数, 分母表示中间路由器与内容提供者总的请求命中的次数. 本指标既表示缓存系统的服务效率, 又可表示中间路由器缓解内容提供者压力的效果.

(4) 平均缓存命中率(average cache hit ratio, ACHR)

缓存命中率反映了每个路由器的缓存命中的情况, 缓存命中率越高, 表示缓存在该路由器中的内容块被利用得越多, 效率越高. 平均缓存命中率即反映了所有路由器的缓存命中率的整体水平, 可作为对缓存服务率的补充. 平均缓存命中率的计算方法为

$$ACHR=\frac{\sum_{i=1}^{r}\frac{hit_i}{hit_i+miss_i}}{r} \tag{28}$$

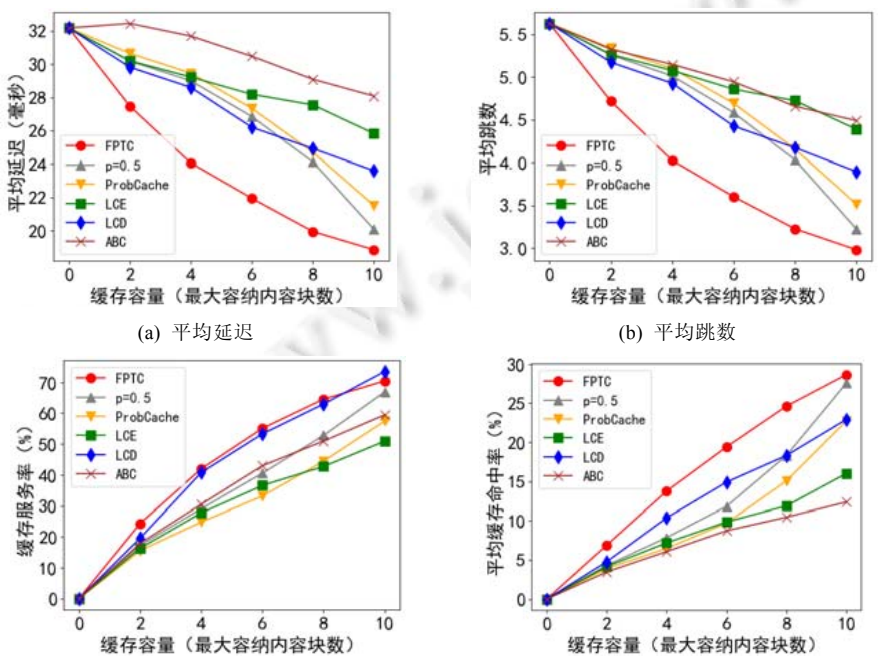
其中, r 表示路由器的数量, hit_i 表示在路由器 i 缓存命中的次数, $miss_i$ 表示在路由器 i 缓存未命中的次数.

4.3 实验结果

4.3.1 缓存容量变化

图5显示了在不同的缓存容量大小下, 不同缓存策略的性能. 图5(a)显示了不同缓存大小下不同缓存策略的平均访问延迟. 如预期所料, 实验中的所有缓存方案的平均访问延迟都随着缓存大小的增加而减少. FPTC策略在内容提供者传回数据后, 就有可能缓存在靠近用户的地方, 从而降低了之后的请求的访问延迟, 因此FPTC 的性能会比较好. 由于 LCE 将导致频繁的缓存替换, 用户的请求在接近自己的路由器上被满足的可能性不大, 因此具有较高的访问延迟. ABC 倾向于将内容缓存在路径中近似中心度最高的节点, 随着时间的推移, 节点间的不均衡性愈发凸显, 中心度高的节点缓存替换更加频繁, 中心度低的节点缓存空间可能会浪费, 中心度最高的节点由于网络拓扑原因可能距离用户较远, 因此访问延迟较高. LCD 只在缓存命中发生的节点返回请求方向的下一跳缓存, 这样虽然会降低缓存的冗余程度, 但是在内容提供者传回数据后, 缓存命中的节点与请求发出的节点之间的距离较远, 需要在命中几个相同的请求后, 缓存所在地才会接近用户, 而此时数据可能已经变得不流行或者不新鲜了. Probabilistic Caching 随机决定是否缓存传入的数据, 有可能快速在靠近用户的节点缓存了数据. 在缓存大小较小时, Probabilistic Caching 的性能接近 LCE; 而当缓存较大时, Probabilistic Caching 的平均访问延迟会有较大的降低. ProbCache 的整体平均延迟高于 Probabilistic Caching,

其平均访问延迟随 *CS* 大小的变化趋势与 Probabilistic Caching 类似, 会随 *CS* 的增大而有较大的降低. 当缓存大小较大时, ProbCache 的平均访问延迟甚至比 LCD 还低. 图 5(b)显示了不同缓存大小下不同缓存策略的平均跳数. 所有缓存方案的平均跳数都随着缓存大小的增加而减少, FPTC 依然表现出几种方案中最佳的性能. 可以看出, 平均跳数与平均访问延迟之间的结果相似. 缓存策略影响跳数和在每个节点的计算开销, 当缓存决策的开销不大时, 跳数与访问延迟的差异就不是特别大. ABC 的平均跳数与 LCE 差别小, 但计算比 LCE 略复杂, 因此平均延迟会略高于 LCE. 图 5(c)显示: 随着缓存大小的增加, 每种缓存策略的缓存服务率均呈上升趋势. 可以看到: 在实验中, FPTC 优于其他缓存方案. 在几种缓存策略中, LCE 的性能最差. 由于 LCE 缓存每个传入内容, 导致缓存会很快填满, 因此内容会迅速从缓存中移出, 这会增加移出的次数并导致缓存未命中, 请求往往需要一直传到内容提供者处才能被满足. LCD 只在缓存命中发生的节点返回请求方向的下一跳缓存, Probabilistic Caching 随机决定是否缓存传入的内容, 这两种方案均能有效降低缓存的冗余, 减少缓存替换的次数, 因此性能会比 LCE 好. ABC 只将内容缓存在路径中近似中心度最高的节点, 一定程度上降低了缓存的冗余度, 且物联网中请求最新内容的概率较大, 因此内容在高中心度的节点中的缓存被满足的可能性较大, 因此缓存服务率也比 LCE 好. ProbCache 也是一种概率缓存, 其缓存服务率随 *CS* 大小的变化趋势与 Probabilistic Caching 类似, 但低于 Probabilistic Caching. 这可能是因为虽然 ProbCache 考虑让内容缓存在越接近用户的地方, 但是较高的缓存概率也使缓存内容被替换的可能性增大, 导致了较多的缓存替换. 由于 LCD 与 Probabilistic Caching 没有考虑具体的传入内容与网络拓扑, 而 FPTC 考虑了内容的流行度, 提高了缓存的内容在未来的一段时间内命中的可能性, 从而减少了缓存的内容替换的次数. 图 5(d)显示, 所有缓存方案的平均缓存命中率都随着缓存大小的增加而减少. 在实验设置的几种缓存大小中, FPTC 的平均缓存命中率比其他几种对比策略要高. 在缓存较小时, FPTC 就已经有了较高的缓存服务率. 可以发现: 平均缓存命中率与缓存服务率的结果有一些不同, 其中差异最大的是 LCD 和 ABC. LCD 使内容缓存在缓存命中发生的节点返回请求方向的下一跳, 这可以使请求到达内容源提供者的概率减小, 但请求命中处离用户较远, 所有路由器收到的总请求次数较多, 导致平均缓存命中率较低. ABC 中中心度低的节点无法缓存内容, 导致这些节点的缓存命中率低, 拉低了整体的平均值.



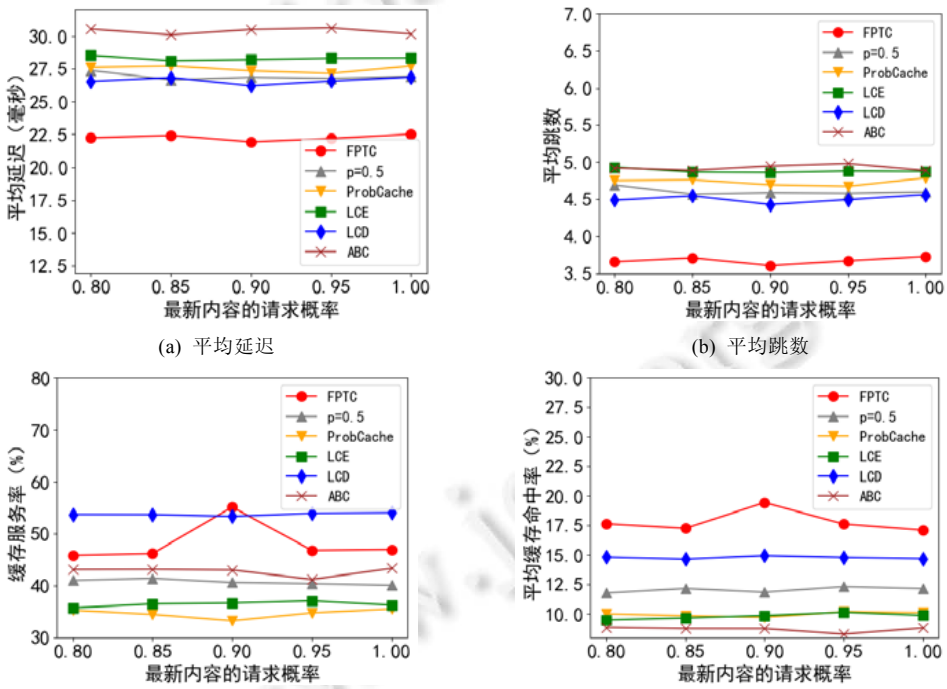
(a) 平均延迟　(b) 平均跳数

(c) 缓存服务率　(d) 平均缓存命中率

图 5　不同缓存容量下性能对比

4.3.2 访问模式中最新内容请求概率变化

图 6 显示了在不同的最新内容请求概率下, 不同缓存策略的性能. 图 6(a)与图 6(b)显示: 几种对比缓存策略的平均访问延迟和平均跳数差别较小, 而 FPTC 在这两个指标上远优于几种对比的缓存策略. 由于 FPTC 通过考虑内容的流行度, 使内容在路由器上的缓存更加有效, 优化了缓存的分配; 此外, 通过对网络拓扑的灵活考虑, 使得内容更加靠近发出请求的用户, 有效降低了平均访问延迟与平均跳数. 同样考虑网络拓扑的还有 ProbCache 和 ABC: ProbCache 使用概率方式进行缓存, 虽然能使内容缓存在靠近用户的路由器, 但同时也增加了路由器上缓存内容的替换, 导致效果不如预期; ABC 虽然将内容缓存在路径上中心度最高的节点, 但由于网络拓扑原因, 中心度高的节点可能离用户较远, 此外, 不同的用户请求的内容不相同, 导致中心度高的节点中的缓存可能无法满足用户, 因此效果也较差. 图 6(c)与图 6(d)显示: 当对最新内容的请求概率变化时, 除了 FPTC 之外, 几种缓存策略的缓存服务率随最新内容的请求概率变化不是特别大, FPTC 的缓存服务率在最新内容请求概率为 0.9 的时候高于 LCD, 其余时候低于 LCD. 在平均缓存命中率方面, FPTC 的平均缓存命中率总是高于 LCD 和其他几种策略. 如前所述, LCD 减小了请求到达内容源提供者的概率, 但请求命中处离用户较远, 需要途经较多的路由器, 增加了所有路由器收到的总请求次数, 从而导致平均缓存命中率较低. FPTC 在保证较高缓存服务率的同时, 使内容缓存处尽可能靠近用户, 因此平均缓存命中率也较高. 整体来说, 最新内容的请求概率在一个较高值范围内变化时对几种缓存策略的影响十分有限.



(a) 平均延迟 (b) 平均跳数

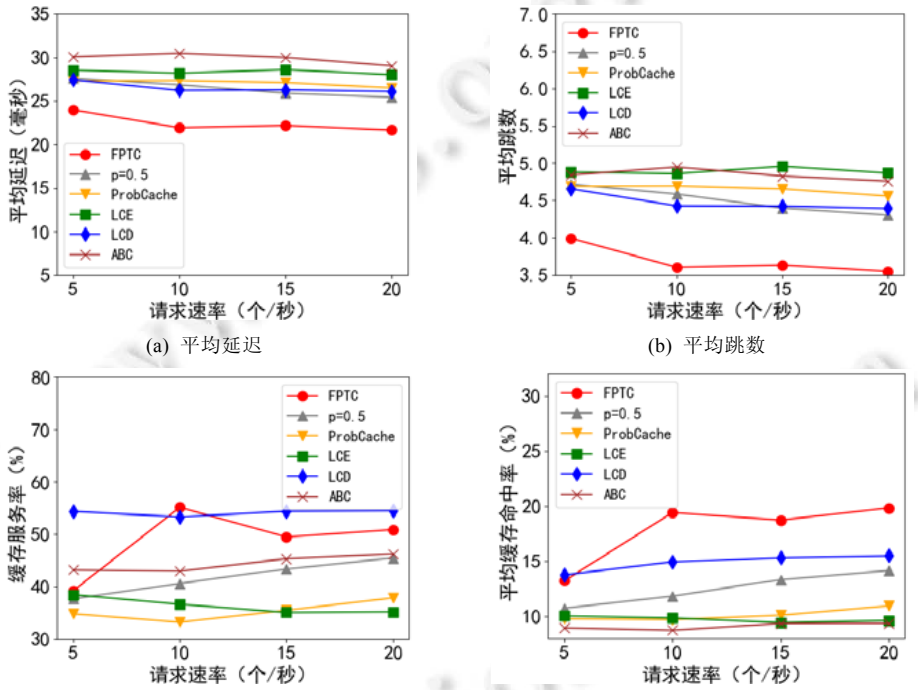
(c) 缓存服务率 (d) 平均缓存命中率

图 6 对最新内容的不同请求概率下性能对比

4.3.3 请求速率变化

图 7 显示了在不同的请求速率下, 不同缓存策略的性能. 图 7(a)与图 7(b)显示: 除 FPTC 外, 几种缓存策略的平均访问延迟与平均跳数随请求速率变化而变化的幅度不大. 由于 FPTC 考虑网络拓扑, 且其使用排名的方式比 ProbCache 和 ABC 更具灵活性, 因此当请求速率增大时, FPTC 会使内容缓存在更靠近用户的地方且缓存替换没有 ProbCache 和 ABC 那么频繁, 从而使平均延迟和平均跳数在请求速率增大时降低更多. 图 7(c)与图 7(d)显示: 随着请求速率的变化, 除 LCE 外, 几种缓存策略的缓存服务率与平均缓存命中率呈上升趋势, LCE 则呈下降趋势. FPTC 的缓存服务率与平均缓存命中率变化幅度较大, 尤其是在请求速率从 5 个/s 到 10

个/s 时的变化很大. 其原因可能是: 请求速率较大时, 路由器记录的请求消息中, 对新内容的请求会更多, 使新内容的流行度比较高, 从而使新内容缓存在路由器中, 后发出的请求在路由器被满足的概率较大, 路由器满足了更大比例的请求, 使缓存服务率与平均缓存命中率上升. LCD 的上升幅度很小, Probabilistic Caching, ProbCache 与 ABC 的上升较 LCD 明显. LCD 由于内容从源提供者处发出后必定会缓存在路由器中, 因此请求速率的增大一定程度上使内容在路由器命中概率增大, 但变化幅度十分有限. 而 Probabilistic Caching 与 ProbCache 使用概率进行缓存, 当请求速率较大时, 由于多次的请求增加了流行内容缓存在路由器中的可能性增加, 同时也会使请求在路由器被满足的可能性增加. ABC 将内容缓存在中心度高的节点中, 当请求速率较大时, 中心度高的节点缓存的内容被满足的次数也会增多. 因此, 这几种缓存服务率与平均缓存命中率呈上升趋势且上升幅度比 LCD 大. LCE 呈下降趋势可能是因为请求速率的上升导致 LCE 的缓存替换次数增加, 请求在路由器命中的次数减少.



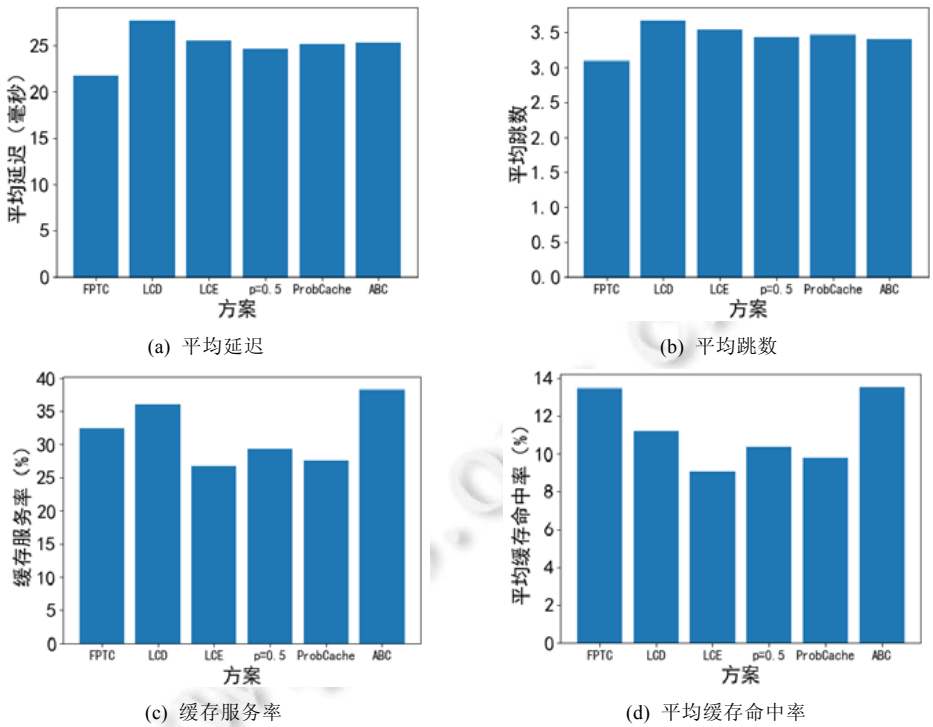
(a) 平均延迟　(b) 平均跳数

(c) 缓存服务率　(d)平均缓存命中率

图 7　不同请求速率下性能对比

4.3.4　网络拓扑变化

图 8 显示了在拓扑 2 下, 不同缓存策略的性能. 可以发现: 在拓扑 2 的结果中, 与拓扑 1 差别较大的是 LCD 和 ABC. 在拓扑 1 中, LCD 几乎是几种对比方案中变现最好的, ABC 的表现则较差; 而在拓扑 2 中, ABC 的性能有大幅提升, LCD 的性能则有较大下降. 文献[31]的研究指出, 网络拓扑结构对缓存策略的性能具有重要影响. LCD 在文献[31]中的 core topology 类型的网络拓扑结构中表现较好, 在 edge topology 类型的网络拓扑结构中表现较差; ABC 则反之. 图 8(a)与图 8(b)显示: FPTC 的平均访问延迟与平均跳数最低; ABC 的平均延迟略高于 Probabilistic Caching 和 ProbCache, 而平均跳数则略低于 Probabilistic Caching 和 ProbCache. LCD 的平均延迟与平均跳数最高, 其次是 LCE. 图 8(c)与图 8(d)显示: 缓存服务率和平均缓存命中率最高的是 ABC; LCD 的缓存服务率高于 FPTC, 但平均缓存命中率低于 FPTC. 另外 3 种对比策略的缓存服务率和平均缓存命中率都较低, 其中最低的是 LCE. 可以发现, FPTC 在两种不同的网络拓扑中都有较好的性能.



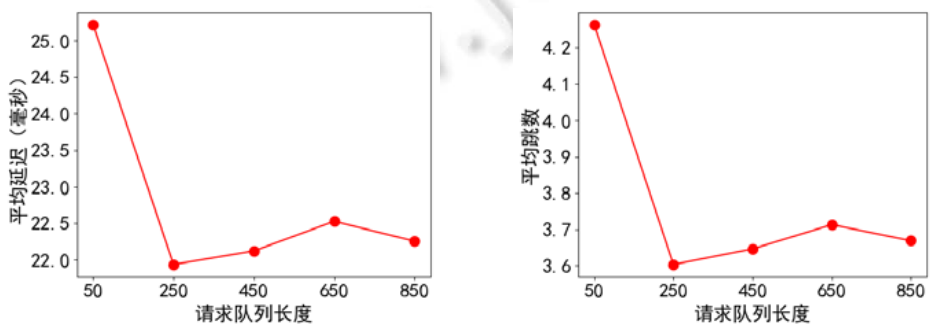
(a) 平均延迟 (b) 平均跳数

(c) 缓存服务率 (d) 平均缓存命中率

图 8 拓扑 2 下的性能对比

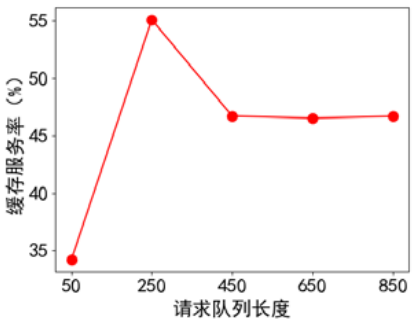
4.3.5 请求队列长度变化

图 9 显示的是所提出的 FPTC 缓存策略中, 请求队列长度不同时对策略性能的影响. 图 9(a)与图 9(b)显示: 平均延迟与平均跳数随请求队列长度的增大首先是降低, 但随后又会有所上升. 图 9(c)与图 9(d)显示: 当请求队列长度较小时, 缓存服务率与平均缓存命中率较低; 随着请求队列长度增大, 缓存服务率与平均缓存命中率首先是上升, 但随后又会下降. 这是因为当请求队列长度过小时, 路由器记录的请求消息太少导致流行度的计算误差较大, 使 FPTC 的性能较差; 当请求队列长度变大时, 路由器的记录会更加全面, 流行度计算更为准确, 从而提高 FPTC 的性能; 但随着队列长度逐渐变大, 路由器的记录包含了较早的请求信息, 这些信息不够新鲜、可用性不大, 且会降低新内容的流行度, 使 FPTC 的性能有所下降. 因此, 请求队列长度过小和过大均会导致 FPTC 性能的降低, 需要选择合适长度的请求队列才能使 FPTC 的性能最大化.

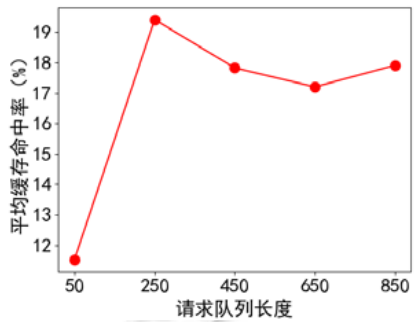


(a) 平均延迟 (b) 平均跳数

图 9 不同请求队列长度下的性能对比



(c) 缓存服务率



(d) 平均缓存命中率

图 9 不同请求队列长度下的性能对比(续)

5 讨论和复杂度分析

所提出的缓存策略 FPTC 同时考虑了数据新鲜度、流行度和网络拓扑，对新内容收益值的预测使用短期预测效果好和运算过程简单的灰色预测模，具有较低的时间和空间开销，整个缓存放置策略简便高效. 本节讨论 FPTC 的时间复杂度和空间复杂度，分析得出其复杂度是可接受的. 算法的复杂度在一定程度上可以反映其计算代价与节点能耗，算法的复杂度越低，则计算代价越小，节点所需能耗也就越低.

5.1 时间复杂度

首先讨论兴趣包处理算法中，对请求队列和统计表的维护操作的时间复杂度. 当兴趣包到达路由器时，对请求队列的操作最多仅有队首元素出队及新元素插入队尾，其时间复杂度为 O(1). 统计表根据请求队列的变化来改变表中条目的请求数和跳数，其时间复杂度也为 O(1). 关于预测方面，假设统计表中共有 m 个条目，统计表需要遍历整张表获得与新条目属于同一生产者且时间戳早于新条目的所有条目，其时间复杂度为 O(m)；假设获得的条目共有 n 条，需要将这 n 个条目按照时间戳进行排序，复杂度为 O($n\cdot\log_2 n$)；在灰色预测中，公式(18)–公式(20)的时间复杂度均为 O(n)；公式(21)中，$2\times n$ 和 $n\times 2$ 的矩阵相乘得到 2×2 的矩阵复杂度为 O(n)，2×2 的矩阵求逆的复杂度为 O(1)，求逆后与 B^T 相乘得到一个 $2\times n$ 的矩阵复杂度为 O(n)，再往下 $n\times 1$ 和 $2\times n$ 的矩阵相乘复杂度为 O(n)；公式(22)和公式(23)的时间复杂度均为 O(n). 因此，预测部分的总时间复杂度为 O(n). 最后是计算条目的收益值部分，对统计表的所有条目进行遍历分类的时间复杂度为 O(m)；假设其中共有 x 种不同的生产者(即分为 x 类)，对于每种分类，即属于同一生产者的所有条目，按照时间戳大小排序的时间复杂度为 O($n\cdot\log_2 n$). 每个条目计算公式(13) –公式(15)和公式(17)的时间复杂度为 O(1). 因此，计算所有条目收益值的总时间复杂度为 O($m+x\cdot n\cdot\log_2 n$). 由于 $x\cdot n\approx m$，因此计算所有条目收益值的总时间复杂度近似为 O($m\cdot\log_2 m$).

综上所述，兴趣包处理算法中，对请求队列和统计表的维护操作的总时间复杂度近似为 O($m\cdot\log_2 m$). 由于请求队列长度有限，m, x 和 n 的大小可以控制在一个较小的范围内，因此整个算法的计算时间较小.

接下来讨论数据包处理算法中，路由器决定是否缓存内容的时间复杂度. 当数据包到达时，假设统计表中共有 m 个条目，则计算排名时只需对整个统计表进行一次遍历即可获得数据包对应的内容排名，其时间复杂度为 O(m)；按照排名决定是否缓存内容的时间复杂度为 O(1). 因此，整个算法的时间复杂度为 O(m)，与前面所述相同. 由于 m 的值较小，整个算法的时间开销较小.

综上所述，FPTC 可以在一个较短的时间内完成，因此其是可接受的.

5.2 空间复杂度

首先讨论请求队列和统计表的空间复杂度. 由于请求队列的最大长度是一个固定的值，因此其空间复杂度为 O(1)；统计表统计请求队列中的每种请求内容数量，其条目数量不会超过请求队列的最大长度.

接下来讨论兴趣包处理算法的空间复杂度. 假设统计表中共有 m 个条目. 在预测时, 统计表首先获得与新条目属于同一生产者且时间戳早于新条目的所有条目, 假设获得的条目共有 n 条, 则此临时数组的空间复杂度为 O(n); 接下来, 公式(18)–公式(23)的空间复杂度均为 O(n), 因此预测部分的总空间复杂度为 O(n). 在计算所有条目的收益时, 对所有 m 个条目分类所需的空间复杂度为 O(m), 计算公式(13) –公式(15)和公式(17)所需空间复杂度为 O(m). 因此, 计算所有条目收益值的总空间复杂度为 O(m). 由于请求队列长度有限, m 和 n 的大小可以控制在一个较小的范围内, 因此整个算法的空间开销较小.

最后讨论数据包处理算法中路由器决定是否缓存内容的空间复杂度. 当数据包到达时, 假设统计表中共有 m 个条目, 只需一个变量即可获得排名, 因此空间复杂度为 O(1), 因此整个算法的空间开销较小.

综上所述, FPTC 所需空间开销不大, 空间复杂度是可接受的.

6 结 论

信息中心网络(ICN)中的网络缓存技术在降低带宽、减少内容检索延迟、降低访问成本、降低网络流量等方面都有显著的改进. 然而, 信息中心物联网中的缓存仍然存在一些挑战, 例如, 如何在数据频繁更新以及物联网用户对数据新鲜程度有要求的环境中缓存内容副本以最大化缓存效率. 本文提出了一种基于内容流行度、网络拓扑、同时考虑了物联网数据更新的缓存策略, 根据内容的流行度、缓存节点与用户之间的距离, 以及新鲜程度来确定是否在节点上缓存相应的内容块. 当内容的流行度与新鲜程度较高且节点距离用户较近时, 节点将缓存该内容. 在 ndnSIM 中对该方案的有效性进行了评估. 实验结果表明: 在不同的缓存大小、对最新内容不同的请求概率和不同的请求速率下, 所提出的策略在平均访问延迟、平均跳数和平均缓存命中率等方面的性能优于现有的缓存策略.

References:

[1] Stryjak J, Sivakumaran M. The mobile economy 2019. GSMA Intelligence, 2019.

[2] Lin C, Jia ZX, Meng K. Research on adaptive future Internet architecture. Chinese Journal of Computers, 2012, 35(6): 1077−1093 (in Chinese with English abstract). [doi: 10.3724/SP.J.1016.2012.01077]

[3] Wang GQ, Huang T, Liu J, *et al*. A new cache policy based on sojourn time in content-centric networking. Chinese Journal of Computers, 2015, 38(3): 472−482 (in Chinese with English abstract). [doi: 10.3724/SP.J.1016.2015.00472]

[4] Wu C, Zhang YX, Zhou YZ, *et al*. A survey for the development of information-centric networking. Chinese Journal of Computers, 2015, 38(3): 455−471 (in Chinese with English abstract). [doi: 10.3724/SP.J.1016.2015.00455]

[5] Jacobson V, Smetters DK, Thornton JD, *et al*. Networking named content. In: Proc. of the 5th Int'l Conf. on Emerging Networking Experiments and Technologies. 2009. 1−12. [doi: 10.1145/1658939.1658941]

[6] Koponen T, Chawla M, Chun BG, *et al*. A data-oriented (and beyond) network architecture. In: Proc. of the 2007 Conf. on Applications, Technologies, Architectures, and Protocols for Computer Communications. 2007. 181−192. [doi: 10.1145/1282380.1282402]

[7] Chai WK, Wang N, Psaras I, *et al*. Curling: Content-ubiquitous resolution and delivery infrastructure for next-generation services. IEEE Communications Magazine, 2011, 49(3): 112−120. [doi: 10.1109/MCOM.2011.5723808]

[8] Dannewitz C, Golic J, Ohlman B, *et al*. Secure naming for a network of information. In: Proc. of the 2010 INFOCOM IEEE Conf. on Computer Communications Workshops. IEEE, 2010. 1−6. [doi: 10.1109/INFCOMW.2010.5466661]

[9] Djama A, Djamaa B, Senouci MR. Information-centric networking solutions for the Internet of Things: A systematic mapping review. Computer Communications, 2020, 159. [doi: 10.1016/j.comcom.2020.05.003]

[10] Abane A, Daoui M, Bouzefrane S, *et al*. A lightweight forwarding strategy for named data networking in low-end IoT. Journal of Network and Computer Applications, 2019, 148: 102445. [doi: 10.1016/j.jnca.2019.102445]

[11] Nour B, Sharif K, Li F, *et al*. A unified hybrid information-centric naming scheme for IoT applications. Computer Communications, 2020, 150: 103−114. [doi: 10.1016/j.comcom.2019.11.020]

[12] Gündoğan C, Pfender J, Kietzmann P, *et al*. On the impact of QoS management in an information-centric Internet of Things. Computer Communications, 2020, 154: 160−172. [doi: 10.1016/j.comcom.2020.02.046]

[13] Din IU, Asmat H, Guizani M. A review of information centric network-based Internet of Things: Communication architectures, design issues, and research opportunities. Multimedia Tools and Applications, 2019, 78(21): 30241−30256. [doi: 10.1007/s11042-018-6943-z]

[14] Gameiro L, Senna C, Luís M. ndnIoT-FC: IoT devices as first-class traffic in name data networks. Future Internet, 2020, 12(11): 207. [doi: 10.3390/fi12110207]

[15] Nour B, Ibn-Khedher H, Moungla H, *et al*. Internet of things mobility over information-centric/named-data networking. IEEE Internet Computing, 2019, 24(1): 14−24. [doi: 10.1109/MIC.2019.2963187]

[16] Sun QB, Liu J, Li S, *et al*. Internet of Things: Summarize on concepts, architecture and key technology problem. Journal of Beijing University of Posts and Telecommunications, 2010, 33(3): 1−9 (in Chinese with English abstract). [doi: 10.3724/SP.J.1238.2010.00585]

[17] Quevedo J, Corujo D, Aguiar RL, *et al*. Consumer driven information freshness approach for content centric networking. In: Proc. of the IEEE INFOCOM 2014—IEEE Conf. on Computer Communications Workshops (INFOCOM WKSHPS). 2014. 482−487. [doi: 10.1109/INFCOMW.2014.6849279]

[18] Laoutaris N, Che H, Stavrakakis I. The LCD interconnection of LRU caches and its analysis. Performance Evaluation, 2006, 63(7): 609−634. [doi: 10.1016/j.peva.2005.05.003]

[19] Yonggong W, Zhenyu L, Qinghua W, *et al*. Performance analysis and optimization for in-network caching replacement in information centric networking. Journal of Computer Research and Development, 2015, 52(9): 2046−2055 (in Chinese with English abstract). [doi: 10.7544/issn1000-1239.2015.20140101]

[20] Laoutaris N, Syntila S, Stavrakakis I. Meta algorithms for hierarchical Web caches. In: Proc. of the IPCCC. 2004. 445−452. [doi: 10.1109/PCCC.2004.1395054]

[21] Psaras I, Chai WK, Pavlou G, *et al*. Probabilistic in-network caching for information-centric networks. In: Proc. of the 2nd ICN Workshop on Information-centric Networking. 2012. 55−60. [doi: 10.1145/2342488.2342501]

[22] Cai L, Wang XW, Wang JK, *et al*. Concept drift learning-based caching strategy in information-centric networks. Ruan Jian Xue Bao/Journal of Software, 2019, 30(12): 3765−3781 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/5621.htm [doi: 10.13328/j.cnki.jos.005621]

[23] Liu WX, Yu SZ, Cai J, *et al*. Scheme for cooperative caching in ICN. Ruan Jian Xue Bao/Journal of Software, 2013, 24(8): 1947−1962 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/4378.htm [doi: 10.3724/SP.J.1001.2013.04378]

[24] Wu HB, Li J, Zhi J. MBP: A max-benefit probability-based caching strategy in information-centric networking. In: Proc. of the IEEE Int'l Conf. on Communications. London, 2015. [doi: 10.1109/ICC.2015.7249222]

[25] Xu A, Tan X, Tian Y. Design and evaluation of a utility-based caching mechanism for information-centric networks. In: Proc. of the 2015 IEEE Int'l Conf. on Communications (ICC). IEEE, 2015. 5535−5540. [doi: 10.1109/ICC.2015.7249204]

[26] Hua Y, Guan L, Kyriakopoulos KG. A Fog caching scheme enabled by ICN for IoT environments. Future Generation Computer Systems, 2020, 111: 82−95. [doi: 10.1016/j.future.2020.04.040]

[27] Song F, Ai ZY, Li JJ, *et al*. Smart collaborative caching for information-centric IoT in fog computing. Sensors, 2017, 17(11): 2512. [doi: 10.3390/s17112512]

[28] Din IU, Hassan S, Almogren A, *et al*. PUC: Packet update caching for energy efficient IoT-based information-centric networking. Future Generation Computer Systems, 2020, 111: 634−643. [doi: 10.1016/j.future.2019.11.022]

[29] Hail MA, Amadeo M, Molinaro A, *et al*. Caching in named data networking for the wireless Internet of things. In: Proc. of the 2015 Int'l Conf. on Recent Advances in Internet of Things (RIoT). 2015. 1−6. [doi: 10.1109/RIOT.2015.7104902]

[30] Jaber G, Kacimi R. A collaborative caching strategy for content-centric enabled wireless sensor networks. Computer Communications, 2020, 159: 60−70. [doi: 10.1016/j.comcom.2020.05.018]

[31] Meddeb M, Dhraief A, Belghith A, *et al*. Cache freshness in named data networking for the Internet of Things. The Computer Journal, 2018, 61(10): 1496−1511. [doi: 10.1093/comjnl/bxy005]

[32] Khedher H, Afifi H, Moustafa H, *et al*. Optimal placement algorithm (OPA) for IoT over ICN. In: Proc. of the IEEE INFOCOM 2017—IEEE Conf. on Computer Communications Workshops (INFOCOM WKSHPS). 2017. 372−377. [doi: 10.1109/INFCOMW.2017.8116405]

[33] Nour B, Sharif K, Li F, *et al*. NCP: A near ICN cache placement scheme for IoT-based traffic class. In: Proc. of the IEEE Global Communications Conf. (GLOBECOM 2018). 2018. 1−6. [doi: 10.1109/GLOCOM.2018.8647629]

[34] Chai WK. Cache "less for more" in information-centric networks (extended version). Computer Communications, 2013, 36(7): 758−770. [doi: 10.1016/j.comcom.2013.01.007]

[35] Pfender J, Valera A, Seah WKG. Easy as ABC: A lightweight centrality-based caching strategy for information-centric IoT. In: Proc. of the 6th ACM Conf. on Information-Centric Networking. 2019. 100−111. [doi: 10.1145/3357150.3357405]

[36] Zhang Z, Lung C, Lambadaris I, *et al*. IoT data lifetime-based cooperative caching scheme for ICN-IoT networks. In: Proc. of the 2018 IEEE Int'l Conf. on Communications (ICC). 2018. 1−7. [doi: 10.1109/ICC.2018.8422100]

[37] Meddeb M, Dhraief A, Belghith A, *et al*. Least fresh first cache replacement policy for NDN-based IoT networks. Pervasive and Mobile Computing, 2019, 60−70. [doi: 10.1016/j.pmcj.2018.12.002]

[38] Zhang GQ, Li Y, Lin T, *et al*. Survey of in-network caching techniques in information-centric networks. Ruan Jian Xue Bao/ Journal of Software, 2014, 25(1): 154−175 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/4494.htm [doi: 10.13328/j. cnki.jos.004494]

[39] Pfender J, Valera A, Seah WKG. Content delivery latency of caching strategies for information-centric IoT. arXiv preprint arXiv: 1905.01011, 2019.

[40] Pfender J, Valera A, Seah WKG. Performance comparison of caching strategies for information-centric IoT. In: Proc. of the 5th ACM Conf. on Information-Centric Networking. 2018. 43−53. [doi: 10.1145/3267955.3267966]

[41] Guo H, Wang X, Chang K, *et al*. Exploiting path diversity for thwarting pollution attacks in named data networking. IEEE Trans. on Information Forensics and Security, 2016, 11(9): 2077−2090. [doi: 10.1109/TIFS.2016.2574307]

[42] Cormen TH, Leiserson CE, Rivest RL, *et al*. Introduction to Algorithms. MIT Press, 2009.

[43] Kayacan E. Grey system theory-based models in time series prediction. Expert Systems with Applications, 2010, 37(2): 1784−1789. [doi: 10.1016/j.eswa.2009.07.064]

[44] Mastorakis S, Afanasyev A, Moiseenko I, *et al*. ndnSIM 2.0: A new version of the NDN simulator for NS-3. Technical Report, NDN-0028, NDN, 2015.

[45] Henderson TR, Roy S, Floyd S, *et al*. NS-3 project goals. In: Proc. of the 2006 Workshop on NS-2: The IP Network Simulator. 2006. 13-es. [doi: 10.1145/1190455.1190468]

[46] Lee B, Pei C, Li F, *et al*. Web caching and Zipf-like distributions: Evidence and implications. In: Proc. of the IEEE Conf. on Computer Communications (INFOCOM'99), the 18th Annual Joint Conf. of the IEEE Computer and Communications Societies: The Future is Now (Cat. No.99CH36320). 1999. [doi: 10.1109/INFCOM.1999.749260]

[47] Rossi D, Rossini G. Caching performance of content centric networks under multi-path routing (and more). Relatório Técnico, Telecom ParisTech, 2011. 1−6.

附中文参考文献:

[2] 林闯, 贾子骁, 孟坤. 自适应的未来网络体系架构. 计算机学报, 2012, 35(6): 1077−1093. [doi: 10.3724/SP.J.1016.2012.01077]

[3] 王国卿, 黄韬, 刘江, 等. 一种基于逗留时间的新型内容中心网络缓存策略. 计算机学报, 2015, 38(3): 472−482. [doi: 10.3724/SP.J.1016.2015.00472]

[4] 吴超, 张尧学, 周悦芝, 等. 信息中心网络发展研究综述. 计算机学报, 2015, 38(3): 455−471. [doi: 10.3724/SP.J.1016.2015.00455]

[16] 孙其博, 刘杰, 黎羴, 等. 物联网: 概念、架构与关键技术研究综述. 北京邮电大学学报, 2010, 33(3): 1−9. [doi: 10.3724/SP.J.1238.2010.00585]

[19] 王永功, 李振宇, 武庆华, 等. 信息中心网络内缓存替换算法性能分析与优化. 计算机研究与发展, 2015, 52(9): 2046−2055. [doi: 10.7544/issn1000-1239.2015.20140101]

[22] 蔡凌, 王兴伟, 汪晋宽, 等. 基于概念漂移学习的 ICN 自适应缓存策略. 软件学报, 2019, 30(12): 3765−3781. http://www.jos.org.cn/1000-9825/5621.htm [doi: 10.13328/j.cnki.jos.005621]

[23] 刘外喜, 余顺争, 蔡君, 等. 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 [doi: 10.3724/SP.J.1001.2013.04378]

[38] 张国强, 李杨, 林涛, 等. 信息中心网络中的内置缓存技术研究. 软件学报, 2014, 25(1): 154−175. http://www.jos.org.cn/1000-9825/4494.htm [doi: 10.13328/j.cnki.jos.004494]

吴海博(1981－), 男, 博士, 副研究员, CCF 高级会员, 主要研究领域为未来互联网,物联网, 缓存技术.

许瑶恭(1998－), 男, 硕士生, 主要研究领域为未来互联网, 缓存技术.

李俊(1968－), 男, 博士, 研究员, 博士生导师, CCF 专业会员, 主要研究领域为互联网体系结构, 互联网安全, 人工智能和大数据应用, 自然科学与社会科学交叉.