软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2019,30(2):381−398 [doi: 10.13328/j.cnki.jos.005572]

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

针对层次化名字路由的聚合机制*

许志伟[1,2], 陈 波[3], 张玉军[2]

[1](中国科学院大学,北京 100049)

[2](中国科学院 计算技术研究所,北京 100190)

[3](Department of Computer Science, University of Memphis, Memphis, TN 38152, USA)

通讯作者: 许志伟, E-mail: xuzhiwei2001@ict.ac.cn

摘 要: 为了从根本上解决现有互联网存在的可扩展性、移动性和安全性等方面的问题,全新的未来互联网体系结构得到了广泛研究.其中,命名数据网络(named data networking,简称 NDN)利用网内缓存和多路转发实现了基于层次化名字的高效数据传输,从根本上解决了现有互联网所面临的问题.内容的层次化名字具有数量庞大、结构复杂等特点,现有的基于 IP 的路由转发机制无法直接应用于 NDN 网络,需要有针对性地研究高效的层次化名字路由机制,保证海量网络内容的正常路由转发.路由聚合是缩减网络路由规模的主要措施.不同于现有的面向本地 NDN 路由表查表过程的优化,路由聚合需要全网协同处理,在不同网络节点上不断对聚合路由进行聚合.这对聚合路由标识和聚合路由可用性评估提出了诸多要求.为此,研究并提出了针对层次化名字路由的聚合机制,包括两个方面的工作:(1) 构建了一种全新的计数布隆过滤器——堆叠布隆过滤器,该过滤器支持多过滤器合并,用于压缩表示被聚合路由名字;(2) 给出了一种动态路由聚合机制,在保证 NDN 网络路由转发准确性的同时,缩小全网路由规模,最大程度地优化了路由转发效率.在真实网络拓扑上构建了仿真平台,经过实验验证,该路由聚合机制以可控的少量冗余转发为代价,有效地压缩了全网路由规模,提升了全网路由转发效率,保证了海量在线内容的高效路由转发,为 NDN 网络投入实际部署提供了前提.

关键词: 层次化名字路由的聚合;可合并计数布隆过滤器;高效计数布隆过滤器查询;可合并压缩表示;动态路由聚合;命名数据网络

中图法分类号: TP393

中文引用格式: 许志伟,陈波,张玉军.针对层次化名字路由的聚合机制.软件学报,2019,30(2):381−398. http://www.jos.org.cn/1000-9825/5572.htm

英文引用格式: Xu ZW, Chen B, Zhang YJ. Hierarchical name-based route aggregation scheme. Ruan Jian Xue Bao/Journal of Software, 2019,30(2):381−398 (in Chinese). http://www.jos.org.cn/1000-9825/5572.htm

Hierarchical Name-based Route Aggregation Scheme

XU Zhi-Wei[1,2], CHEN Bo[3], ZHANG Yu-Jun[2]

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

[2](Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China)

[3](Department of Computer Science, University of Memphis, Memphis, TN 38152, USA)

Abstract: TCP/IP-based Internet is faced with severe challenges in scalability, mobility, security, and controllability, which makes it necessary to research the clean-slate architecture for Internet. Named Data Networking (NDN) leverages in-network caching and multi-

* 基金项目: 国家自然科学基金(61672500, 61572474); 国家重点研发计划(2016YFE0121500)

Foundation item: National Natural Science Foundation of China (61672500, 61572474); National Key Research and Development Program of China (2016YFE0121500)

收稿时间: 2017-05-17; 修改时间: 2017-08-15; 采用时间: 2018-03-15; jos 在线出版时间: 2018-04-16

CNKI 网络优先出版: 2018-04-16 10:59:55, http://kns.cnki.net/kcms/detail/11.2560.TP.20180416.1059.009.html

path transmission to support data name-based packet transmission, and conquers the aforementioned challenges faced by the conventional Internet. Considering the huge number and the significant diversity of hierarchical names used in NDN, the existing IP-based routing cannot be directly applied, and efficient hierarchical name-based routing mechanisms are desired to gurantee stable data transmission. Route aggregation is the core mechanism for FIB size reduction in the TCP/IP-based Internet. Hierarchical name-based route aggregation leverages router cooperation over the whole network to aggregate routes recursively, different from the existing researches for name-based route lookup optimization on a single router. To achieve effective hierarchical name-based route aggregation, identifier generation is focused on in route aggregation and utility evaluation of the aggregated routes. Ultimately, a hierarchical name-based route aggregation scheme is proposed, including two parts: (1) A novel combinable counting Bloom filter for representing aggregated route names compactly, namely, Compounded Counting Bloom Filter (CCBF); (2) A dynamic route aggregation scheme over the whole network for aggregating routes recursively according to their utility. To evaluate the performance of the proposed route aggregation scheme, extensive simulations are performed based on real network topology. Evaluation results show that the proposed scheme can significantly reduce FIB size and improve FIB lookup efficiency at the cost of small number of false positive lookup results. This work provides a foundation for practical NDN deployment.

Key words: hierarchical name-based route aggregation; combinable counting Bloom filter; efficient membership query in counting Bloom filter; combinable compact representation; dynamic route aggregation; named data networking

现有的互联网在可扩展性、移动性和安全性等方面面临严峻的挑战[1−3].互联网的主流应用形式已经从基本的端到端通信转变为大规模内容传输,现有互联网端到端的设计原则无法适应这些新兴的内容密集型应用形式,加剧了互联网的内容传输压力.同时,随着移动互联网的发展,由移动终端产生的流量占总体网络流量的比重日益增加,而 TCP/IP 体系结构面向常连接设计,IP 地址被赋予身份和位置双重功能,无法有效支撑移动性应用的要求.

为了从根本上解决互联网面临的这些问题,需要采用 clean-slate 方式设计并建设全新的未来互联网[3−5].命名数据网络(named data networking,简称 NDN)[6,7]根据信息名字请求和传输数据,不再依赖固定的 IP 地址传输数据,保证了互联网的可扩展性和移动性;同时,采用网内数据泛在缓存和多路传输,有效地保障了这种基于内容名字的传输机制的效率.鉴于 NDN 网络的这些优点,命名数据网络的设计思想已经得到了广泛的重视和研究,将会应用于包括 5G 在内的新兴网络体系[8,9].

NDN 网络基于内容名字进行路由,内容名字由不定数量的任意字符串组成.与 IP 地址相比,内容名字的结构更为复杂,因此,现有的互联网路由机制无法直接应用于 NDN 网络.同时,当前互联网拥有海量的在线内容,相应的内容名字数量也极其庞大.综合考虑 NDN 网络路由内容名字的复杂性及其庞大的数量,基于名字的路由技术将是影响 NDN 网络效率的最为关键的问题,将直接影响 NDN 网络的效率和可用性[1].名字是内容的标识,目前常见的内容名字可以分为两类.

- 一类是以 NDN 为代表的层次化名字,采用可读的文本方式组织,一般由多个“/”隔开的名字段(字符串)组成,名字段的数量和长度都是任意的,类似于现行互联网中的 URL(uniform resource locator)[7].
- 另一类是以数字签名为代表的扁平化名字.这类名字是经过散列(hash)得到的散列值,具有很好的安全性[2].各类扁平化名字具有其专门的编码方式,与 IP 地址类似,配备有完整的命名和路由规则.

当前,关于层次化名字路由的研究还不成熟,尤其是如何优化层次化名字路由,还没有完善的解决方案.

目前,关于层次化名字路由传输的研究主要集中在提升路由表的名字前缀(名字前缀由多个名字段构成)查找速度方面[10−14],例如,通过前缀树压缩和 GPU 并行加速等方法来提高路由查表速度.现有的 NDN 路由机制为了获取特定内容,需要针对该内容的名字添加相应的路由表项,利用内容名字标识路由,并关联可以用于转发的接口信息(不唯一),因此,NDN 的路由表规模与现有 TCP/IP 网络的路由表规模相比将出现巨大的增长.另一方面,NDN 网络中内容来源多样,信息可以来自多信息源,或者移动信息源,甚至以存储在网内缓存中的信息副本作为信息来源,导致网络内容版本变化更加频繁,进一步加剧了层次化名字路由器的负载.因此,仅靠提升单节点查表速度难以保证海量在线内容的高效路由转发.针对这些问题,我们需要约减 NDN 网络整体路由规模,实现层次化路由的聚合和聚合更新,保证高效、准确地转发内容请求包及相应的内容包,提升 NDN 网络整体效率.

路由聚合是降低路由表规模、提升查表效率和优化路由转发效率的主要措施.路由聚合用一条聚合路由代

替大量原始路由,从而减小路由表规模[15].NDN 网络各节点路由聚合过程如图 1 所示,聚合前,路由表中存在各个内容对应的表项;聚合后,具有公共前缀和相同转发接口的表项被聚合为一条聚合路由.不同于现有无类域间路由利用子网号(IP 前缀)作为聚合路由标识,NDN 网络聚合路由的标识应包括两部分:首先是内容名字前缀,其次还需要利用后缀聚合标识区分这一前缀下发往不同接口内容名字.这一聚合路由标识上的差异实际上反映了 IP 路由和层次化名字路由的本质差异.IP 地址路由标识了路由转发的目的地址,特定子网内 IP 地址有限,可以用子网号代表这些地址,作为聚合路由标识.NDN 网络中,拥有同一名字前缀的路由的指向不确定,仅利用名字前缀无法代表所有被聚合的路由,形成的聚合路由也无法作为路由转发的真正依据,需要引入名字后缀的压缩表示来共同标识被聚合的路由.

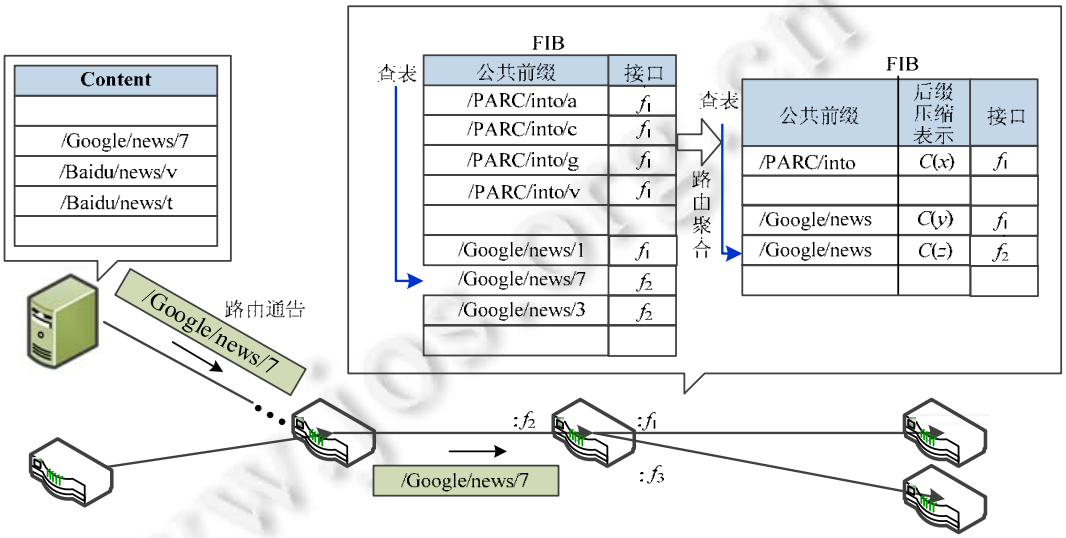

Fig.1 An example for hierarchical name-based route aggregation

图 1 层次化名字路由的聚合实例

为了实现全网路由聚合,最大程度地减少名字路由条目数量,在完成特定前缀的名字路由聚合后,需要将被聚合路由通告给其他网络节点,以便这些节点进一步进行路由聚合,与本地路由表压缩的效果对比,路由聚合能够在网络层面真正大幅度缩减路由规模.

为了约减全网路由规模,提升层次化名字路由查表及转发效率,本文对层次化名字路由聚合问题进行了系统的分析,采用名字前缀和后缀压缩表示共同标识路由,提出了支持聚合的后缀压缩表示机制,并以路由可用性为依据构建了高可用的路由聚合机制,为 NDN 网络投入实际部署提供了前提.主要包括如下工作.

(1) 对层次化名字路由聚合问题进行了研究,明确了将名字后缀纳入聚合后的路由标识的必要性.

(2) 为了支持聚合路由在节点间的交换及进一步的聚合,本文提出了一种支持更新及合并操作的后缀压缩表示机制,堆叠计数布隆过滤器.

(3) 为了限制后缀压缩表示引起的冗余转发,我们依据聚合路由的可用性构建了一套动态名字路由聚合机制,在保证路由转发正确性的前提下,最大程度地聚合名字路由,约减全面路由规模,提升了路由转发效率,保证了海量在线内容的高效路由转发.

本文首先在第 1 节对问题背景和相关研究进行系统分析.第 2 节分析层次化名字路由问题.第 3 节提出一种可合并计数布隆过滤器,用于名字后缀的压缩表示,和名字前缀一同标识聚合路由,以便进行本地路由聚合和全网聚合路由通告.在此基础上,本文给出一套高可用的动态路由聚合机制.最后,本文在真实网络拓扑的基础上构建仿真平台,验证所提出的层次化名字路由聚合机制.

1 背景及研究现状

1.1 NDN网络

NDN 根据层次化的名字而不是 IP 地址路由和转发数据包.为了从 NDN 网络中获取需要的内容,一个请求节点 Consumer 首先发送请求包(interest),其中包括被请求内容的层次化名字.网络中的路由节点收到该请求包后,分 3 步进行处理.

(1) 首先查找 PIT 表(pending interest table).PIT 表记录了之前转发过的所有未被响应的请求包信息,包括被请求内容的名字、收到该请求的时间、接口及转发接口.如果在 PIT 表中找到了特定名字的请求信息,则意味着已经转发过该请求,将新的请求到达端口加入该表项的接口列表,收到内容后一并处理;如果没有找到,则建立新的 PIT 表项.

(2) 查找该节点的内容缓存 CS(content store),如果缓存了该内容,则直接构建相应的内容包(content),并从收到请求的接口返回内容包.

(3) 如果本地未缓存所请求内容,则查找转发表 FIB(forwarding information base),从指定接口将该请求转发给下一跳的各个邻居.如果传输路径上的节点都没有缓存被请求内容,则请求包最终将被转发到内容发布者 Producer,由内容发布者构建内容包.内容包中将包括内容名字、内容、内容包签名及签名信息(发布者公钥等).然后,同样由内容发布者从收到请求包的接口返回该内容包.内容包由转发请求包的路由节点按照其对应 PIT 表项中的记录沿请求到达接口原路径返回,最终达到内容请求者 Consumer.

各节点 FIB 中记录全网所有在线内容的转发信息,路由条目数量巨大,为了保证正常的基于层次化名字的路由转发,必须有效地约减路由条目数量,提升路由查表及转发效率.

1.2 相关研究分析

针对现有的 TCP/IP 网络规模不断增长的问题,大量工作试图通过 IP 路由聚合和压缩加以解决[15,16].其中,基于无类域间路由的 TCP/IP 网络路由聚合机制利用子网号(IP 前缀)代表被聚合的子网 IP 地址,有效地约简了全网路由规模[15],保证了现有 TCP/IP 网络体系结构的互联网的可用性.其后,为了进一步缩短路由查表时间,提升路由转发效率,出现了大量本地路由表压缩方面的研究.Degermark 等人通过一种全新的 hash 机制实现了路由 IP 前缀的压缩[16].文献[17]首次使用基本的 Bloom 过滤器实现了路由表查表过程的优化.针对 IPv4 和 IPv6 路由查表过程,文献[18]提出了一种编码机制,在保证查找效率的同时,降低了查找过程中的误判率.然而,由于名字路由不再基于 IP 实现路由,而是以内容名字为路由标识进行路由,路由标识的数量不可控,且结构更为复杂,这些研究都无法直接应用于层次化名字路由.

在 NDN 路由转发方面,张丽霞等人[7]进行了大量基础性研究.首先,在文献[19]中提出了 NDN 网络的自适应多路转发机制,在转发路径选择方面优化了网络传输效率;其次,文献[20,21]分别提出了 NDN 网络中基于内容名字的最短路径优先协议 OSPFN 和链路状态协议 NLSR,实现了基本的层次化名字路由.在所提出的这些关键问题的引导下,层次化名字路由机制得到了更为广泛的研究[22−24],包括分层路由和路由可达性评估等机制被先后引入层次化名字路由过程.为了进一步提升路由查表效率,文献[10,13,14,25]通过哈希表、名字前缀的前缀树索引和前缀树压缩等方式优化了 NDN 路由节点路由查表(内容名字查找)效率.文献[11]通过 GPU 提升了前缀树查找效率.为了实现板卡级别的包过滤,进而优化 NDN 网络内容获取效率,文献[12]利用布隆过滤器实现了网络节点本地 FIB 表、PIT 表和 CS 缓存的表项板卡级缓存,提升了转发效率.然而,当今互联网已经得到了全方位的应用,在可以预期的互联网扩大应用的大背景下,网络信息数量将进一步急剧增长[3],仅仅通过改进本地路由查表过程、提升查表效率,并不能从根本上解决未来层次化名字路由在应用中的路由规模问题,需要通过研究包括路由聚合在内的全网路由规模约减机制来保证正常的路由查表及转发.

2 针对层次化名字路由的聚合过程分析

构建在现有互联网层次化网络拓扑之上的 TCP/IP 路由聚合机制,通过无类域间路由将子网 IP 地址聚合成对应的子网网络号以约减路由表规模,并将聚合后的路由通告给其他网络节点,以便进行进一步的聚合,最终形成高度聚合的聚合路由,在全网范围内约减路由规模.NDN 的层次化名字路由根据路由名决定从哪几个接口转发内容请求.下面通过一个例子分析名字路由聚合过程.注意,为了简化分析,假设各节点路由的转发接口均一致,不再特别说明.参照 IP 地址聚合机制,层次化名字聚合过程如图 2(a)所示.

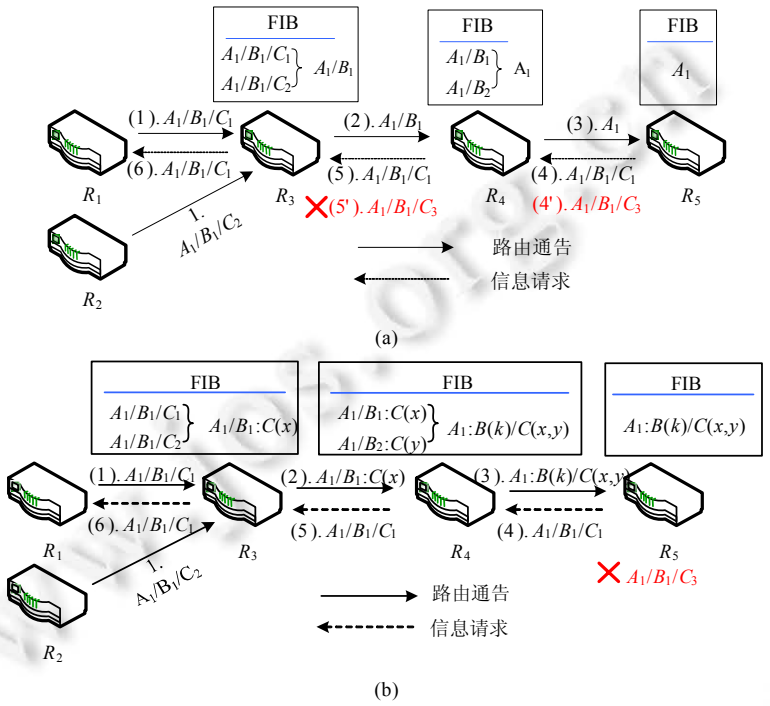

Fig.2 Potential problems in hierarchical name-based route aggregation

图 2 针对层次化名字的路由聚合过程的潜在问题

(1) 在路由建立过程中,路由器 R_1 和 R_2 分别向路由器 R_3 发送了针对名字 $A_1/B_1/C_1$ 和 $A_1/B_1/C_2$ 的路由通告.

(2) 路由器 R_3 收到这两条路由通告后,聚合成路由表项 A_1/B_1 并添入其 FIB(fowarding information base)表.

(3) 路由器 R_3 继续向周边路由器通告聚合后的路由表项,最终路由器 R_4 将收到的路由通告 A_1/B_1 与其本地路由 A_1/B_2 进一步聚合,得到路由 A_1 并通告给 R_5.

(4) 在数据请求阶段,路由器 R_5 向 R_4 发送请求包,请求名字为 $A_1/B_1/C_1$ 的内容.

(5) 路由器 R_4 会根据路由表项 A_1/B_1 将请求包转发给路由器 R_3.

(6) 最终由路由器 R_3 转发给路由器 R_1,路由器 R_1 返回相应的数据内容给请求者.

从上述过程可知,采用层次化名字聚合能够缩小全网路由表规模,进而提升路由查找效率.然而,如果仅仅采取与 IP 地址类似的聚合方式,那么当请求没有聚合过的内容时,这种聚合方式将会出现问题.还是在上述例子中,如果在步骤(4′)中,R_5 请求名字为 $A_1/B_1/C_3$ 的数据,那么在步骤(5′)中,R_4 同样会基于聚合后的路由表项将请求包转发给路由器 R_3,但因为路由器 R_3 中并没有聚合名字为 $A_1/B_1/C_3$ 的路由信息,将导致数据请求失败.

从上述实例可以看出,名字聚合不能完全采用 IP 路由聚合的方式.原因在于,IP 路由地址是一种具有固定长度的由 0 和 1 组成的比特串,当聚合后的 IP 地址前缀长度为 k 时,其可能包含的被聚合后缀的取值最多只有 2^{32-k} 种,IP 地址聚合时考虑到了所有可能的后缀取值,聚合后通过 IP 前缀信息标识路由;与 IP 地址不同,内容名字的

取值可以是任意字符串.层次化名字的一个名字段在理论上有无穷多种取值,如图 2(a)中,名字前缀“A_1/B_1/”后面可以添加各种字符串(名字后缀),因此,以名字前缀为标识的聚合结果无法囊括所有可能的名字后缀,即使请求名字同路由名字的前缀吻合,也无法保证可以根据该路由转发请求并获得对应的内容.因此,在名字路由聚合过程中,不仅需要抽离出待聚合路由的公共前缀来作为新的聚合路由前缀,同时为了准确地将多条路由聚合为一条聚合路由,也需要生成这些待聚合路由名字后缀的压缩表示,和聚合路由前缀一起标识聚合路由,如图 2(b)所示.为了便于节点间交换聚合路由并进一步聚合这些路由,实现全网路由聚合,后缀压缩表示不仅需要能够准确地压缩表示被聚合路由的名字后缀,还需要支持多个后缀压缩表示的合并,以便构建新的聚合路由.如图 2(b)中,R_3完成对路由 $A_1/B_1/C_1$和 $A_1/B_1/C_2$的聚合后,会将聚合路由 A_1/B_1:$C(x)$通告周边节点 R_4,其中,$C(x)$为名字后缀 C_1和 C_2的压缩表示,R_4可以将收到的聚合路由 A_1/B_1:$C(x)$和本地路由 A_1/B_2:$C(y)$进一步聚合,压缩表示 B_1,B_2的同时,合并 $C(x)$,$C(y)$,形成新的聚合路由 A_1:$B(k)/C(x,y)$,聚合路由标识由名字前缀 A_1 和名字后缀压缩表示 $B(k)/C(x,y)$共同构成.

考虑到全网海量内容名字聚合后的巨大熵空间,过度的聚合必将带来路由查表过程中的误判,因此在名字聚合过程中,需要保证聚合后路由的可用性.在此基础上,最大程度地聚合路由,缩减全网路由的规模.如何构建准确的后缀压缩表示机制,并以聚合路由可用性为依据完善层次化名字路由的聚合机制,以更少的路由条目表征正确的路由转发路径,这将是一个极具挑战性的问题,也是层次化名字路由聚合过程中必须解决的一个关键问题.

3 针对层次化名字路由的聚合机制

根据上一节的分析,为了构建针对层次化名字路由的聚合机制,主要需要解决两个问题.

(1) 聚合路由标识问题,具体来说就是名字后缀的压缩表示问题;

(2) 以路由可用性为基准的路由聚合问题.

为了解决这些问题,本节首先对聚合过程中的核心问题——名字后缀压缩表示问题进行分析,并按照分析结论设计支持路由名字后缀压缩表示的计数布隆过滤器——堆叠计数布隆过滤器,和名字前缀一起构成聚合路由标识;最后,在对 NDN 路由可达性分析的基础上构建一套高可用动态路由聚合机制.

3.1 后缀压缩表示问题分析

网络内容不断发生变化,存在内容源失效、内容过期和内容版本更新等情况,为了能够动态地添加并更新指向相关内容的聚合路由,后缀压缩表示机制不仅要表示特定内容后缀的存在性,同时还要支持特定内容后缀的删除和更新.经过多年的发展,布隆过滤器(Bloom filter,简称 BF)已经成为应用最为广泛的压缩表示机制,衍生出了多种不同类型的布隆过滤器,其中,计数布隆过滤器(counting Bloom filter,简称 CBF)被设计用来同时支持添加操作和删除操作[26].CBF 通过将 BF 的单比特位单元扩展为用于计数的固定长度的多比特位单元,记录各单元被重复设置为 1 的次数(所添加名字段的哈希值命中这一单元的次数),以便哈希到这些单元上的已添加元素需要被删除时,通过减小这些单元上的计数器值来实现元素删除.除了用 l 比特的计数器替代基本布隆过滤器的单比特单元外,CBF 完全等同于 BF,可以按照基本布隆过滤器的配置方法配置,包括配置过滤器单元(计数器)数、哈希函数数量等.CBF 的哈希函数命中特定单元的最大次数 c 大于特定整数 i 的概率具有上界[27],可以根据这一上界配置 CBF 各单元计数器大小(比特位串长度 l).通常情况下,4 比特长的计数器即可满足构建 CBF 的需要,以极大的概率保证任一单元的计数器都不会超出其最大计数范围.CBF 可以满足路由聚合过程中路由更新引起的动态添加、删除名字后缀的需要.为了优化 CBF 的性能,在支持添加操作和删除操作的同时,降低存储开销和假阳性率,多种新型计数布隆过滤器先后被提出来.其中,Cohen 等人提出了 Spectral Bloom Filter (SBF)[28],可以确保其任一单元的计数器都不会超出其最大计数值,但是与 CBF 相比增加了的存储开销.文献[29]基于 d-left 哈希提出 dlCBF,与 CBF 相比,在同样的假阳性率下优化了存储开销.MPCBF[30]利用多级比特数组在保证查询效率的基础上实现了 CBF 的功能,其存储开销和假阳性率与 CBF 相比都有所改善.

但是,如果利用现有各类计数布隆过滤器压缩表示路由名字后缀,在全网路由聚合过程中,来自不同节点的

相关路由后缀的压缩表示将被进一步聚合,这就需要将后缀压缩表示(计数布隆过滤器)合并,而现有的各类计数布隆过滤器[27−30]都不支持多个计数布隆过滤器的合并操作.现有计数布隆过滤器都通过类似计数器的机制来记录哈希函数重复命中各个单元的情况,多个计数布隆过滤器合并时可以选择的合并算子包括加运算、或运算和异或运算.首先,逐个单元进行计数器值的按位或运算和异或的运算结果不能代表不同元素命中各单元的计数总和,因此无法用来支持计数布隆过滤器的合并.而对应单元的计数器相加同样也存在问题.例如,

(1) 将计数布隆过滤器的计数器长度设为 4 比特,最大计数值为 15.

(2) 当一个节点 R 周边的 16 个节点中的一个 R_j 压缩表示一个后缀名字段 nsg_x 后,nsg_x 的 k 个哈希结果指向的计数布隆过滤器 CBF_j 的计数器的值 $c_{1j},\ldots,c_{kj}$ 均加 1,得到$\forall c_{ij}\geqslant 1, i\in\{1,2,\ldots,k\}, j\in\{1,2,\ldots,16\}$.

(3) 假设 R 周边的 16 个节点都压缩表示了 nsg_x,当 R 需要进一步聚合这条路由时,需要合并来自 16 个邻居的计数布隆过滤器 $CBF_1,\ldots,CBF_{16}$.如果采用相加的方式合并这些 CBF,则需要将这些计数布隆过滤器各单元计数器的值相加,即 $c_i=c_{i1}+\ldots+c_{i16}$.由于$\forall c_{ij}\geqslant 1$,故 $c_i\geqslant 16$,超出了计数器的最大计数范围.

注意,这与上述计数器最大值的概率上界并不冲突.文献[27]中上界的证明条件是计数布隆过滤器用于压缩表示不重复元素,如果不同路由存在相同的名字后缀,计数布隆过滤器的计数器很快就会溢出.究其原因,现有计数布隆过滤器在压缩表示被添加的元素的过程中,没有保留足够的信息,无法发现两个压缩表示结果中相同的元素,合并后压缩表示结果中会出现重复元素.综上所述,由于现有计数布隆过滤器不能有效地排除不同路由节点上记录的相同名字后缀的影响,因此不能支持来自不同路由节点的名字路由的聚合操作.为了实现支持路由聚合的名字后缀压缩表示机制,并和路由名字前缀一同标识聚合路由,需要构建一种全新的压缩表示机制,该机制需要同时支持:(1) 后缀名字段的动态添加和删除;(2) 多个压缩表示结果的合并.针对这两个需求,我们在第 3.2 节中设计了一种全新的计数布隆过滤器——堆叠计数布隆过滤器(compounded counting Bloom filter,简称 CCBF),用于被聚合路由名字后缀的压缩表示.

3.2 堆叠计数布隆过滤器

为了支持名字后缀的动态添加和删除,同时支持多个压缩表示的合并,我们在现有计数布隆过滤器思想的基础上构建的新的后缀压缩表示机制需要:

(1) 归并重复后缀名字段的添加,支持多个压缩表示的合并;

(2) 记录后缀名字段哈希结果命中过滤器各个单元的次数,支持后缀名字段的动态添加和删除.

通过加运算合并现有计数布隆过滤器时,由于无法排除记录在多个计数布隆过滤器中的重复后缀名字段,造成合并结果中会出现计数器计数溢出的情况.我们注意到,基本布隆过滤器可以通过按位或归并重复名字段.这是由于在基本布隆过滤器中重复添加和合并重复名字段时,重复的名字段的所有哈希结果都将命中过滤器中同一个单元,即使重复置 1,也不会改变 BF 比特数组的取值,从而屏蔽了重复名字段的影响.例如,将一个布隆过滤器 BF_1 同另外一个布隆过滤器 BF_2 合并的过程中,如果 BF_1 和 BF_2 中都添加了后缀名字段 nsg_x,nsg_x 的 k 个哈希结果分别指向的这两个布隆过滤器的 k 个单元(位),$b_{11},\ldots,b_{k1}$ 和 $b_{12},\ldots,b_{k2}$,全部被置 1.注意到,合并 BF_1 和 BF_2 后,仍有:

$$\left.\begin{aligned} & b_i = b_{i1}\cup b_{i2} \\ & \text{s.t. } i\in\{1,2,\ldots,k\} \end{aligned}\right\} \tag{1}$$

合并结果中同样是这 k 个单元被置 1,可以有效地归并记录在不同布隆过滤器中的重复名字段.根据这一观察,我们利用基本布隆过滤器可以归并重复元素的特点,通过堆叠多个相同的基本布隆过滤器来构建支持合并的计数布隆过滤器.多个基本布隆过滤器的特定位置的单元可以共同记录该位置被所添加的后缀名字段的哈希结果命中的次数(每次将其中一个基本布隆过滤器的相应单元置 1),从而构建了一种全新的支持合并的计数布隆过滤器——堆叠计数布隆过滤器.CCBF 通过伪随机发生器使用多个基本布隆过滤器,在有效使用所包含的过滤器的同时,保留了各单元被重复置 1 时过滤器的使用顺序.

CCBF 的数据结构如图 3 所示.

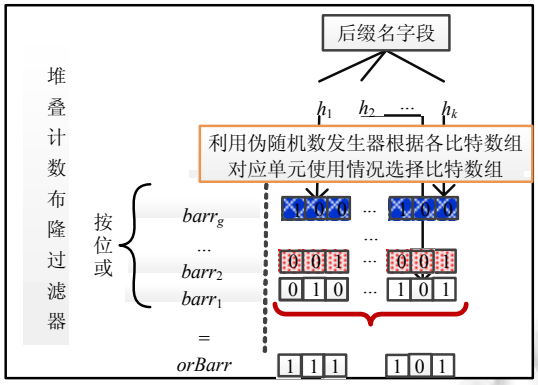

Fig.3 Framework of compounded counting Bloom flitter

图 3 堆叠计数布隆过滤器的结构

CCBF 由 g 个比特数组(长度为 m)和它们按位或的结果数组 $orBarr$ 构成.其中,g 的大小以能够满足计数需要为基准,本节后续将具体说明.orBarr 是 CCBF 的 g 个比特数组按位或的结果,用于提升 CCBF 的查询效率.CCBF 除了支持名字段的添加、删除和查询外,还支持多个 CCBF 的合并.下面对这些操作逐一加以说明.

(1) 添加、删除和查询后缀名字段

CCBF 在添加一个名字段 nsg 的过程中,针对 k 次哈希操作中的第 j 次哈希操作,利用伪随机数生成器,根据哈希结果对应位置单元(下标为 $hash_j(nsg)$的单元)已经被置 1 的比特数组的数量,随机选中 g 个比特数组中的一个比特数组 $barr_i$(CCBF 的第 i 个比特数组),将其下标为 $hash_j(nsg)$的单元置 1.同时,如果为名字段 nsg 的 k 个哈希结果选择的比特数组满足:

$$\left.\begin{array}{l}\forall barr_i[hash_j(nsg)]=1\\ \text{s.t. } j\in\{1,2,...,k\}\end{array}\right\} \tag{2}$$

则说明该名字段已经添加过,放弃本次添加.注意,本文采用的伪随机数发生器根据第 $hash_j(nsg)$个单元已经被置 1 的比特数组的数量来选择下一个要使用的比特数组,具体操作就是针对各个单元随机生成一个数组标号的排列,并以这些排列为列构建矩阵,在产生伪随机数的过程中,以特定单元已经被置 1 的数组的数量加 1 为行号,查找特定单元对应的列上相应行中记录的数组标号,返回该标号代表的数组.因此,选中的比特数组对应的单元必然还没有被置 1.这样既避免了名字段的重复添加,又高效地实现了特定单元被置 1 次数的计数,为删除名字段提供了前提.具体添加操作的计算过程如算法 1 所示.算法 1 中,利用伪随机数发生器随机选择比特数组,针对一个特定的哈希结果指向的单元,总能在 g 个比特数组中找到一个数组,其对应单元为 0(证明见定理 1),可以持续支持新数据的插入.

算法 1. *CCBF.add*(*nsg*).

输入:*nsg*. //待压缩表示的后缀名字段

1 **Begin**

2 *barrSet*=*CCBF.GetAllBarr*()

3 **For** *j*=0 **to** *CCBF.k*–1 //为哈希结果选中待写入的比特数组,如果不是重复添加,则写入,其中 k 是哈希函数的数量

4 $p_j=Hash_j(nsg)$

5 $Barr_i$ =*RandChoice*(*barrSet*,p_j,*Counter*(p_j)) //用伪随机数发生器根据 p_j 对应单元已经被置 1 的比特数组个数(*Counter*(p_j)),随机选择 1 个比特数组

6 $Barr_i[p_j]=1$

7 **End For**

8 *UpdateorBraa*() //g 个比特数组按位或得到更新后的 *orBarr*

9 **End**

定理 1. CCBF 过滤器中包含 g 个比特数组,令这些比特数组第 p_j 个单元被置为 1 的个数为 c_j,则 c_j 大于等于 g 的概率可控,具有概率上界.

证明:令 CCBF 能够插入的名字段的最大值为 n,其相应的比特数组长度为 m,哈希函数个数为 k,比特数组数量为 g,则 g 个比特数组的第 p_j 个单元被置为 1 的数量 c_j 大于等于 g 的概率为

$$\Pr(\max(c_j)\geqslant g)\leqslant m\binom{nk}{g}\left(\frac{1}{m}\right)^g \tag{3}$$

根据斯特林公式化简可得:

$$\Pr(\max(c_j)\geqslant g)\leqslant m\sqrt{\frac{2\pi nk}{4\pi^2(nkg-g^2)}}\frac{nk^{nk}}{(nk-g)^{nk-g}g^g}\left(\frac{1}{m}\right)^g \tag{4}$$

由文献[26]中的公式:

$$m=\frac{nk}{\ln 2} \tag{5}$$

可知,在布隆过滤器的最优配置中,$n\times k$ 与 m 成线性关系,因此,$n\times k$ 远大于 g,公式(4)中根号内的值小于 1.同时,对第 2 项放大,得到公式(3)的一个松弛上界:

$$\Pr(\max(c_j)\geqslant g)\leqslant m\left(\frac{enk}{gm}\right)^g \tag{6}$$

将公式(5)代入可得:

$$\Pr(\max(c_j)\geqslant g)\leqslant m\left(\frac{e\ln 2}{g}\right)^g \tag{7}$$

为 g 个比特数组特定位 p_j 被置为 1 的次数 c_j,得到 g 的概率上界. □

根据公式(7),当 g 为 16 时,CCBF 任意一位全部被置 1 的概率为 $1.37\times10^{-15}\times m$,在通常网络应用中,这种情况不会出现,即当向包括 16 个比特数组的 CCBF 添加名字段时,总可以为 k 个哈希结果找到对应单元为 0 的比特数组,完成添加操作.

下面对添加操作的计算复杂度进行分析和比较.算法 1 共进行了 k 次哈希、k 次随机数组选择及 k 次数组读写操作,最后将 g 个比特数组按位或得到更新后的 *orBarr*,计算复杂度为 $O(k\times H_1+k\times H_2+k+m\times g)$,其中,$H_1$ 为布隆过滤器哈希过程的时间开销,H_2 为伪随机数生成器选择数组的时间开销,伪随机数生成器实际上就是在针对各单元的随机数组标号排列中查找一个元素,其计算复杂度远低于布隆过滤器的哈希函数,而比特数组按位或操作的复杂度 $m\times g$ 同样小于布隆过滤器的哈希函数的复杂度,因此,本算法的计算复杂度规约为 $O(k\times H_1)$.现有的计算布隆过滤器,包括 CBF、SBF、dlCBF 和 MPCBF,添加新元素时直接查找并更新对应的计数单元,而本算法还要通过额外的伪随机生成器选择计数单元对应的比特数组,虽然计算复杂度均为 $O(k\times H_1)$,但算法 1 的计算开销将近似 2 倍于上述 4 类计数布隆过滤器的添加操作的计算开销.当然,鉴于现有布隆过滤器哈希函数的高效性[26],CCBF 添加操作的总体计算开销仍较小,不会影响 CCBF 的应用.

布隆过滤器作为一种压缩表示机制,除了可以节省存储空间外,最关键的是可以提升后缀名字段存在性查询的效率.与添加和删除操作相比,后缀名字段(路由)查询是路由聚合机制中最常见的操作,为了优化聚合路由(CCBF 的名字段)查询效率,我们在 CCBF 中添加了一个记录 CCBF 中 g 个比特数组按位或的结果的数组 *orBarr*,在进行查询的过程中,无需逐一查看 g 个比特数组,判断哈希结果对应单元是否为 1,而是直接查看 *orBarr* 的对应单元是否为 1 即可,具体查询操作如算法 2 所示,查询操作的复杂度完全等价于基本布隆过滤器查询操作的复杂度,与 CBF 和 MPCBF 的查询过程类似,可以直接定位各个哈希函数对应的单元并进行查询,具有相同的计算复杂度.由于 SBF 在查询过程中在通过哈希函数定位计数单元后仍需要通过索引最终找到这些计数单元,以完成查找,dlCBF 需要在 d 个分区中逐一定位并查找各个哈希对应的单元,存在重复操作.因此,

SBF、dlCBF 查询效率低于 CBF、MPCBF 以及算法 2.

算法 2. *CCBF.query*(*nsg*).

输入:*nsg*. //待查询的后缀名字段

```
1  Begin
2    For j=1 to CCBF.k   //k 是哈希函数的数量
3      p_j=Hash_j(nsg)
4      If CCBF.orBarr[p_j]≠1
5        Return false
6      End If
7    End For
8    Return true
8  End
```

CCBF 删除操作的计算过程类似于添加操作的计算过程,这里不再赘述,仅就其中的关键操作步骤进行说明.首先确认该名字段可以删除(类似算法 2 的查找过程);然后,通过伪随机数生成器定位上一次添加操作中各个哈希函数操作的单元所在的比特数组,将这些比特数组对应单元清零,并更新 *orBarr*,完成名字段删除操作.因此,CCBF 删除操作的计算复杂度与添加操作类似,为 $O(k\times H_1)$.与其他计数布隆过滤器相比,CCBF 的删除过程需要通过伪随机数发生器找到需要进行单元清零操作的比特数组,因此,删除操作的计算开销近似 2 倍于其他计数布隆过滤器的删除操作的开销.

(2) 合并操作

不同于其他现有计数布隆过滤器,CCBF 支持合并操作,即将多个配置完全相同的 CCBF 合并为 1 个,以便应用于类似路由聚合这样需要归并已有名字段的场景.多个 CCBF 的合并操作相当于将这些 CCBF 压缩表示的名字段(添加的元素)合并,即等同于在一个 CCBF 中添加其他 CCBF 中压缩表示过的名字段.CCBF 中的伪随机数发生器可以确保哈希结果对应单元置 1 时选择的比特数组有固定的顺序,CCBF 中添加后缀名字段(元素)的顺序不会影响各个比特数组的取值,详见定理 2.因此,按照不同 CCBF 比特数组的标号,可以按顺序合并各个比特数组(按位或,与 BF 的合并操作相同),实现多个 CCBF 的合并.

当然,多个 CCBF 合并后压缩表示的名字段数量仍然受到其比特数组数量 g 和长度 m 的限制.以两个 CCBF 的合并过程为例,合并过程主要是将两个 CCBF 中的 g 个比特数组两两按位进行或操作,同时将两者的 *orBarr* 也按位或.具体如算法 3 所示.

算法 3. *CCBF.combine*(*other*).

输入:*other*. //待合并的另外一个 CCBF 过滤器,两个 CCBF 的配置完全相同,且合并后的后缀数没用超过 CCBF 的容量 *n*

```
1  Begin
2  If  CCBF.EstSize()+other.EstSize()≥CCBF.n   //判断合并后压缩表示的名字段数量是否已经超过了 CCBF 的容量 n
3    Return error
4  End If
5  barrSet=CCBF.GetAllBarr()
6  otherBarrSet=other.GetAllBarr()
7  For j=0 to CCBF.g−1
8    barr=barrSet.first
9    otherbarr=otherBarrSet.first
10   barr.Or(otherbarr)
```

```
11      barrSet.Remove(barr)
12      otherBarrSet.Remove(otherbarr)
13  End For
14  CCBF.orBarr.Or(other.orBarr)
15  End
```

其中,为了保证合并后不会超过 CCBF 的配置容量 n,CCBF 利用 estSize 函数估计当前已添加的名字段的数量(已添加名字段数量).CCBF 每次添加 1 个名字段时会有 k 位被置 1,因此可以利用 g 个比特数组中 1 的总数除以 k,得到已经添加的名字段的数量,estSize 函数的计算过程如公式(8)所示。

$$estSize=\frac{\sum_{j=0}^{g-1}barr_j.Sum4ones()}{k} \tag{8}$$

其中,g 和 k 为 CCBF 配置的比特数组数量和哈希函数数量,比特数组的 sum4ones 函数可以计算它其中被置 1 的单元的数量.算法 3 的计算过程包括估计已添加名字段总数和合并比特数组两部分,其中,

- 估计已添加名字段的复杂度为 $O(m\times g)$,其中,m 为比特数组长度,g 为比特数组数量(最大计数范围);
- 合并比特数组的复杂度也是 $O(m\times g)$.

因此,算法 3 总的计算复杂度为 $O(m\times g)$.注意,现有其他计数布隆过滤器均无法支持合并操作.

定理 2. CCBF 中添加元素的顺序不会影响 CCBF 中各个比特数组的取值.

证明:对于一组元素 $nsg_1,\ldots nsg_i,\ldots,nsg_n$,将其按顺序添加到一个 CCBF 中.不失一般性,假设有两个名字段——nsg_x 和 nsg_y,其哈希结果命中了 CCBF 的第 j 个单元,CCBF 中的伪随机数发生器以确定顺序在 $barr_1,\ldots,barr_g$ 中选中两个比特数组,将其第 j 个单元置 1.这一过程中,随机数发生器选中的下一个比特数组的依据是当前第 j 个单元被置 1 的比特数组的个数.因此,如果首先添加 nsg_x,第 j 个单元被置 1 的比特数组的个数为 0,随机数发生器选中 $barr_{(1)}$,将 $barr_{(1)}[j]$置 1.当添加 nsg_y 时,选中 $barr_{(2)}$,将 $barr_{(2)}[j]$置 1.同样,如果先添加 nsg_y,将对 $barr_{(1)}[j]$置 1,然后添加 nsg_x 时,将对 $barr_{(2)}[j]$置 1.无论 nsg_x 和 nsg_y 的添加顺序如何,添加结果都是 $barr_{(1)}[j]$和 $barr_{(2)}[j]$被置 1.对特定单元的多次加 1 操作的顺序不影响该单元使用,因此,CCBF 中添加名字段的顺序不会影响 CCBF 中各个比特数组的取值. □

综上所述,堆叠计数布隆过滤器可以高效地完成名字段的添加、删除和查询,其计算复杂度基本与现有计数布隆过滤器等同,其中,路由过程中最为频繁的查询操作的效率优于或等同于其他计数布隆过滤器.同时,堆叠计数布隆过滤器 CCBF 支持高效的多过滤器合并,为聚合路由后缀合并(及其他需要合并多个压缩表示的场景)奠定了基础.CCBF 空间复杂度为 $O(m\times g)$,与 CBF 的空间复杂度 $O(m\times\log_2 g)$相比,使用的存储空间更多.同样地,与 dlCBF 和 MPCBF 相比也需要更多的存储空间(除 SBF 使用的存储空间随添加元素增长外,dlCBF 和 MPCBF 都在 CBF 的基础上优化了存储空间开销[29,30]),但是根据定理 1,g 的取值在实际应用中具有上界(通常情况下为 16),因此,CCBF 空间复杂度可控,在当前网络设备存储空间不断扩大的背景下,完全可以应用于各类互联网应用场景.在假阳性率方面,CCBF 中哈希函数定位数组单元的方式与 BF 和 CBF 相同,函数个数 k 和数组长度 m 的配置也与 BF 和 CBF 相同[26],因此,CCBF 中添加一个元素时,k 个哈希结果均发生碰撞的概率(假阳性率)与 BF 和 CBF 完全相同.虽然 CCBF 的假阳性率高于优化了假阳性率的 SBF、dlCBF 和 MPCBF,但是路由聚合过程中,假阳性的出现只会增加冗余转发数量,不会引起转发错误,并非路由聚合过程中的首要因素,不会影响 CCBF 在聚合路由后缀压缩表示中的应用.

综上所述,CCBF 的主要设计目标是实现多个 CCBF 的合并,以便支持路由聚合操作.在此基础上,CCBF 优化了其查询效率,为构建高效的路由聚合机制提供了前提.在基于 CCBF 的后缀压缩表示的基础上,可以将指向同一接口的多条具有公共名字前缀的路由聚合为 1 条聚合路由,用公共名字前缀和后缀压缩表示共同标识聚合路由(如图 1 所示),有效地避免了第 2 节中指出的仅采用路由名字前缀来标识路由时出现的路由错误,构建了准确、高效的路由标识.

3.3 基于路由可达性的动态名字路由聚合机制

在上述层次化名字聚合路由标识的基础上,本文利用路由可达性度量聚合路由可用性,并作为路由聚合条件,将多条稳定可达的具有公共名字前缀的路由(包括其他节点通告的路由)聚合为一条聚合路由;当可达性下降后,进行解聚合,还原被聚合路由,降低聚合程度,提升路由可达性;聚合和解聚合都需要通告周边邻居节点.按照这一动态聚合方式,本文构建了一套基于路由可达性的动态名字路由聚合机制.

本文之所以将路由可达性作为路由聚合的条件,其原因是聚合路由压缩表示了被聚合路由的名字后缀(或后缀压缩表示),将不可避免地在路由查找过程中引入假阳性,将没有聚合过的路由纳入聚合路由查找结果,进而在 NDN 网络中带来冗余的转发.由于 NDN 网络采用多路转发机制,因此假阳性引起的冗余转发不会影响正常的内容获取,仅仅会加大网络负载.当然,过大的网络负载同样会影响网络效率,甚至会抵消路由聚合后路由规模缩减带来的效率提升.如果假阳性问题引起的网络负载过大,将会进一步引起网络拥塞等问题,因此,我们需要最大程度地保证转发的正确性,同时,通过路由聚合减小路由规模,提升网络效率.其中,为了回滚聚合路由,需要在低速存储中备份被聚合路由,在减小路由节点高速内存空间的同时,实现了基于路由可达性的动态聚合过程.本文名字路由聚合记录如图 4 所示.

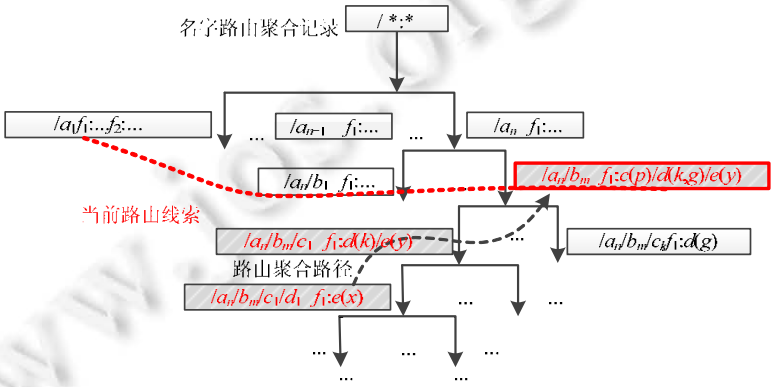

Fig.4 An example for dynamic route aggregation

图 4 动态路由聚合过程实例

NDN 网络节点的不同接口可以获取不同内容,因此,本文的聚合机制在聚合相关路由的同时,保留了各个接口的独立记录.为每条聚合路由针对不同接口设置独立的后缀压缩表示单元和相应的可达性统计单元,形成聚合路由的标识二元组结构:⟨路由前缀;转发接口信息⟩,其中,转发接口信息中记录了接口 ID、路由后缀压缩表示、转发延迟统计和可达性统计等状态信息(如图 4 所示).可达性统计为 *RP*(*faceID*),其中,*faceID* 为相应的接口 ID.可达性统计值为最近转发出去的请求的响应比,代表路由可用性的统计结果,如公式(9)所示.

$$RP(faceID)=\int\frac{n_s(t)}{n(t-1)}\mathrm{e}^{-t+1}\mathrm{d}t \tag{9}$$

其中,$n(t-1)$为 $t-1$ 时刻总共发出的内容请求包数量,$n_s(t)$为 t 时刻收到的内容响应包数量.其中,通过卷积运算淘汰旧的统计信息,提升当前请求响应情况在统计结果中的比重.针对接口 *faceID*,*RP*(*faceID*)准确地反映了依据聚合路由从该接口转发请求、获取内容的可能性,可以作为评估聚合路由可用性的标准.根据文献[31]中的相关定理,当节点间连通概率大于等于 $\lg(N)/N$ 时(N 为网络节点数),网络为强连通网络.因此,我们可以将这一条件作为判断路由可达性的阈值,如果所有 *RP*(*faceID*)都大于 $\lg(N)/N$,则可以获取分布在任意网络节点上的内容.

当一组具有公共前缀的聚合路由在所有接口上的可达性都达到了阈值,且持续了从某个时刻 t 起的两个时刻时,则这些路由被认为是稳定的路由,可以和其他具有公共前缀的稳定路由一同进一步聚合新的聚合路由.这种延迟决策机制可以避免聚合过程中的抖动现象(频繁重复聚合和解聚合的循环).同样地,聚合路由两个时刻

内持续不可用,则需要将其分解为之前被聚合的路由,以提升路由准确性,改善路由可达情况.为了支撑聚合路由回滚,需要在各个节点保留如图 4 所示的线索树结构,在聚合路由的同时保留被聚合路由信息,以便更新和回滚聚合路由.其中,当前路由线索是当前使用的聚合路由的线索,当需要更新和回滚路由时,可以快速找到相应的聚合路由,并回滚或更新相关被聚合路由.本文的动态聚合机制可以在保证路由可用性的前提下最大程度地聚合路由,缩小全网路由规模,提升层次化名字路由效率.

4 实验评估

4.1 方案部署与实现

我们在 NDN 网络的仿真平台 ndnSIM[32]的基础上实现了本文针对层次化名字路由的聚合机制.

- 首先,我们对 ndnSIM 的 FIB 模块进行了修改.现有的 ndnSIM 中,每条路由包含路由名字前缀和转发接口信息.基于本文动态路由聚合机制,我们在转发接口信息中添加了后缀压缩表示 C 和可达性统计 RP(见公式(9)).RP 中的统计值 $n(t-1)$是在查表转发请求时计数,而 $n_r(t)$是下一个时刻收到的内容包的数量.根据不同的预定容量 n,用于压缩表示后缀名字段的 CCBF 按布隆过滤器最优配置设置比特数组长度 m 及哈希函数个数 k[26],同时,根据本文第 3.2 节的分析,比特数组数量被设置为 16.在此基础上,FIB 模块周期性更新 RP,根据更新过的 RP 分析现有路由的可用性,将持续可用的路由进行进一步聚合,得到新的聚合路由,并周期性地通告周边节点.同时更新在低速外部存储中的记录路由聚合过程的数据结构(如图 4 所示),以备聚合路由回滚时还原其中备份的路由信息.
- 其次,我们修改了转发模块 ndn-forwarding-strategy.实现了基于名字前缀和后缀压缩表示的转发机制.

仿真平台在本地计算机上部署,其配置如下:Ubuntu 13.04,内核版本 3.8.8,Intel Core i7 3.4G CPU,DDR3 1866 16G 内存,1TB 硬盘.为了保证仿真结果真实、可靠,我们采用了一个真实的网络拓扑——欧洲 EBONE 网络拓扑[33],包括 279 个节点、731 条边,其中包括 65 个边界网关节点、45 个网关节点和 169 个边缘路由节点.在边缘节点中,我们随机选择了 10 个节点作为内容发布节点 producer,实验中请求的各类内容全部由这 10 个节点发布.另外,选择了 30 个节点作为内容请求方 consumer,内容请求服从参数α为 0.6 的 Zipf-Mandelbrot 分布,即被请求的多数内容为热门内容,其他内容很少被请求.为了全面验证本文路由聚合机制的有效性,实验中使用了多个不同规模的数量集,这些数据集的具体信息见表 1.

Table 1 Datasets

表 1 数据集

数据集名称	squidGuard blacklist	Shalla's blacklists	Fabrice Prigent's blacklist	MESD blacklists	Alexa top 1M
大小(KB)	27 796	9 791	8 449	8 457	9 787
条目数	4 583 626	1 700 000	1 682 512	1 342 541	1 000 000

在此基础上,通过对这些数据进行组合,我们可以得到各种规模的名字数据集.在其基础上,我们对本文路由聚合机制的有效性进行了全面的评估.首先比较了路由聚合前后的路由表规模;其次,通过与 NDN 原生路由查表机制以及利用 CBF 压缩路由表的对比方案比较,给出了路由聚合前后的路由查表效率的比较;最后,通过对比分析了本文路由聚合机制带来的假阳性问题.

4.2 聚合前后路由表规模对比分析

路由聚合的目标是减少路由条目数量,从而提升路由查表效率,进而优化网络传输效率.同时,路由条目的减少还会减少路由设备内存的使用,降低 NDN 网络部署成本.为了验证本文路由聚合机制的聚合效果,我们在数据集 Alexa top 1M 上比较了各个路由节点在同一时刻采用聚合后的路由条目数量和不采用聚合的情况下路由条目数量的比值 R,在 2 000s 内对 279 个路由器每隔 6s 记录一次 R 值,仿真结果如图 5 所示.

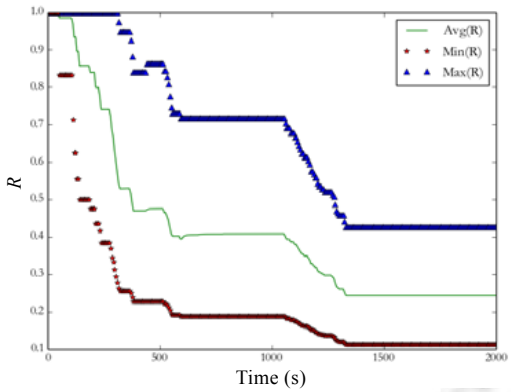

Fig.5 Effection on FIB size reduction of route aggregation

图 5 路由聚合对路由表规模的影响

仿真过程中,随着时间 Time 的推移,路由聚合程度不断提升,路由规模不断缩小,在 1 400s 附近,全网路由聚合达到稳定状态.全部 279 个节点中,在全网路由聚合完成后,路由条目数量至少压缩了 60%,平均压缩了 75.47%,路由聚合对路由规模的缩减效果明显.注意,由于现有数据集中多数名字(URL)只有 2 级,如果应用于实际部署的 NDN 网络中,路由名字层次变化更多,路由聚合效果也会更为显著.

4.3 聚合路由效率分析

作为影响网络传输效果的关键因素,路由查表效率直接影响网络数据包转发延时.路由聚合主要通过减少路由条目数量(路由表规模)来提升路由查表效率,进而缩小网络数据包转发延时.利用本文路由聚合机制,可以将多条具有公共前缀的层次化名字路由聚合为单条聚合路由,并利用这些路由的公共名字前缀和后缀压缩表示来共同标识该聚合路由.因此,与未进行聚合的层次化名字路由相比,本文的路由聚合机制在缩短名字前缀查找时间的同时,也引入了路由后缀的查找过程.为了验证本文聚合过程对路由查表时间的影响,我们首先针对不同规模 N_s 的名字路由(1 万、10 万和 100 万)进行路由聚合,并比较聚合前后各路由的查表时间总和 LTime.实验结果如图 6(a)所示,圆形点线代表未聚合路由查表时间总和,方形点线为本文聚合路由查表时间总和.可以看出,未被聚合的路由的查表时间与路由规模成正比例关系,随着路由规模 N_s 的增长线性增加;而聚合后路由查找基本不受路由规模增长的影响,仅表现出 1s 的差异,如图 6(a)所示.究其原因,主要是由于聚合后减小了前缀数量及前缀查找时间;同时,本文在后缀压缩表示查询方面的优化(为 CCBF 添加 16 个比特数组按位或得到的归并结果 *orBarr*)也避免了大量后缀查询时延的引入.

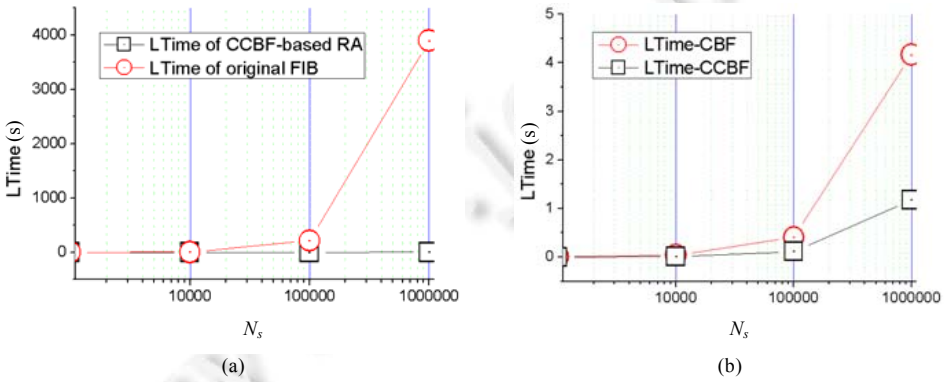

Fig.6 Comparison of route lookup time

图 6 路由查找效率对比

为了进一步验证本文路由聚合机制的查表效率,我们在 CBF 的基础上构建了一套路由压缩机制作为对比方案,采用与本文的聚合路由相同的前缀和 CBF 压缩表示的路由后缀共同标识压缩后的路由.压缩后的路由规模与本文路由聚合后的路由规模相同,通过这一方式对比验证本文聚合路由查表效率.如图 6(b)所示,方形点线为 CBF 的查询时间 LTime-CBF,圆形点线为本文 CCBF 的查询时间 LTime-CCBF.在不同规模的路由表上,本文方案的查表时间始终只有对比方案的一半,本文建立在 *orBarr* 上的查询过程的效率要优于 CBF 的查询效率.

同时,我们对比了聚合前后路由更新效率(如图 7 所示),路由更新时间等于所有路由的查找、删除和添加操作的总和,用 UpTime 代表.图 7(a)中,圆形点线为未聚合路由总更新时间,方形点线为本文方案路由总更新时间.未聚合路由的整体更新时间都随路由规模 N_s 的增长呈线性增长,本文路由聚合机制的更新时间增加较慢,路由规模对路由更新效率的影响同样存在.为了进一步评估本文机制的路由更新效率,图 7(b)中,我们对比了本文机制的路由更新时间和基于 CBF 的对比方案的路由更新时间.圆形点线为本文机制的总路由更新时间,方形点线为对比方案的总路由更新时间.两个方案的总路由更新时间都随路由规模 N_s 的增长呈线性增长,本文路由更新时间略小于对比方案路由更新时间的 2 倍,虽然与路由查表时间相比,本文的路由聚合机制的路由更新时间较长,但路由更新不是网络路由转发过程中最关键的操作,操作频率较低,且独立处理,基本不影响路由查表过程.

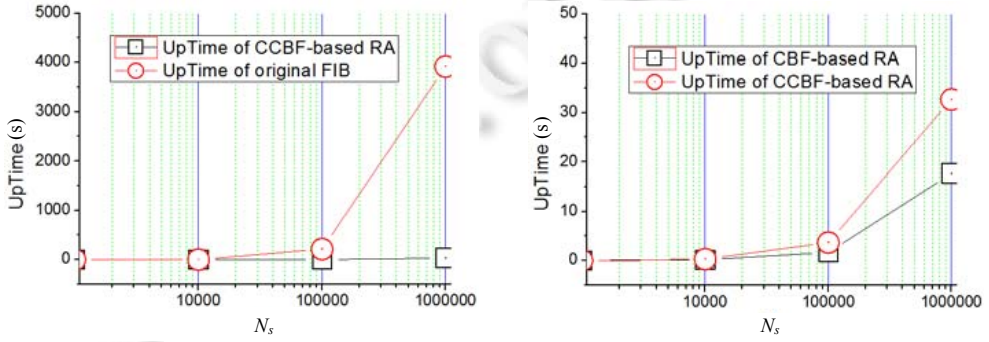

Fig.7 Comparison of route updating time

图 7 路由更新时间对比

4.4 假阳性分析

本文的聚合标识在使用路由名字前缀的同时,还引入了基于CCBF的后缀压缩表示.CCBF 本身可能会在查询过程中引入假阳性,即查询到未被压缩表示过的名字段.假阳性在路由聚合问题的背景下,将会引发路由查询的假阳性误判,将数据包发往没有通告过相关路由的接口.注意,假阳性不会影响数据包从通告过该路由的接口正常转发,仅仅是增加了冗余的路由转发.本节实验验证了本文后缀压缩表示所采用的 CCBF 布隆过滤器的假阳性情况,并与基于 CBF 的对比方案的假阳性进行了对比.在布隆过滤器的配置过程中,本文为各个过滤器设定的最大假阳性率均为 0.01.图 8 中,菱形点线为 CCBF 在不同规模数据集下的查询假阳性数 Fp-CCBF,方形点线为对比方案在不同规模数据集下的查询假阳性数 FP-CBF.两种布隆过滤器在不同规模的数据集上的假阳性值相同.

为了进一步分析假阳性的出现情况,分析两种方案在不同的添加比例 R_{fill} 下的假阳性变化情况,本文针对 3 种不同规模的数据集(1 万、10 万和 100 万)进行了实验.其中,$R_{\text{fill}}=n_c/n$,n 为布隆过滤器容量(分别取 1 万、10 万和 100 万),n_c 为添加到布隆过滤器的名字段数量.如图 9 所示,在不同规模的数据集下,两种方案的假阳性查询数量在 R_{fill} 达到 0.85 后出现了假阳性查询,并在 R_{fill} 到达 1 时达到最大值,两种方案的假阳性查询数量始终保持相同.本文方案在查询假阳性方面等同于基于计数布隆过滤器 CBF 的对比方案,没有带来额外的假阳性查询.

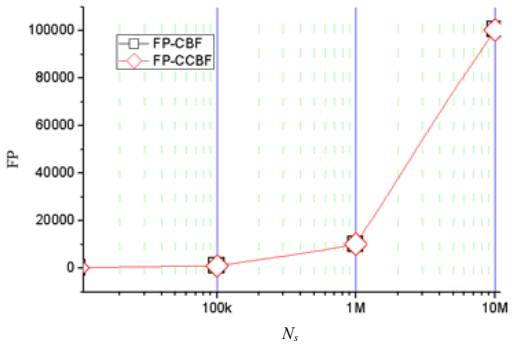

Fig.8 False positive under datasets with different size

图 8 不同规模数据集下的假阳性

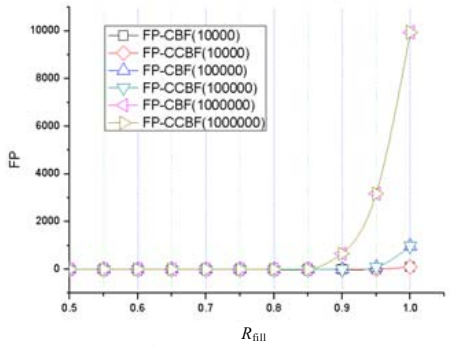

Fig.9 False positive with different R_{fill}

图 9 不同 R_{fill} 下的假阳性

5 结 论

为了实现针对层次化名字路由的聚合机制,进而优化 NDN 等信息中心网络的传输效率,本文首先分析了层次化名字路由的聚合问题,明确了路由聚合过程中需要同时利用路由名字前缀和后缀压缩表示来共同标识聚合后的路由.为了实现对名字后缀的压缩标识,同时在路由聚合过程中通告和进一步聚合(合并)这些后缀压缩标识,本文提出了一种全新的计数布隆过滤器 CCBF.CCBF 在支持多过滤器合并的同时,还优化了查询效率.在此基础上,本文进一步提出了面向路由可达性的动态路由聚合机制,在保证路由可达性的前提下最大化聚合路由.我们在真实网络拓扑及数据集上构建了实验环境,验证了本文路由聚合机制的有效性,本文路由聚合机制通过有效聚合路由,以较小的冗余路由转发(假阳性路由查询)为代价,减小了路由规模,优化了网络传输效率,为 NDN 等信息中心网络投入实际应用提供了前提.

本文的路由聚合程度由路由可达性动态决定.作为影响可达性的关键因素,如果能够有效地降低 CCBF 的假阳性查询结果,则不但可以避免过多的冗余转发,而且可以提升路由可达性,进一步提升路由聚合程度,减少路由条目数量.为此,在下一步的研究工作中,应该将综合分析现有计数布隆过滤器的优化思路,降低堆叠计数布隆过滤器的查询假阳性,进一步改进路由后缀压缩表示机制及相应的路由聚合过程.

References:

[1] Bari MF, Chowdhury SR, Ahmed R, Boutaba R, Mathieu B. A survey of naming and routing in information-centric networks. IEEE Communications Magazine, 2012,50(12):44–53. [doi: 10.1109/MCOM.2012.6384450]

[2] 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 2007 Conf. on Applications, Technologies, Architectures, and Protocols for Computer Communications. New York: ACM Press, 2007. 181–192. [doi: 10.1145/1282427.1282402]

[3] Roberts J. The clean-slate approach to future internet design: A survey of research initiatives. Annals of Telecommunications, 2009, 64(5):271–276. [doi: 10.1007/s12243-009-0109-y]

[4] Xie GG, Zhang YJ, Li ZY, Sun Y, Xie YK, Li ZC, Liu YJ. A survey on future internet architecture. Chinese Journal of Computers, 2012,35(6):1109–1119 (in Chinese with English abstract). [doi: 10.3724/SP.J.1016.2012.01109]

[5] Wu C, Zhang YX, Zhou YZ, Fu XM. 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]

[6] Jacobson V, Smetters DK, Thornton JD, Plass MF, Briggs NH, Braynard RL. Networking named content. In: Proc. of the 5th Int'l Conf. on Emerging Networking Experiments and Technologies. New York: ACM Press, 2009. 1–12. [doi: 10.1145/1658939.1658941]

[7] Zhang LX, Jacobson V, Zhang BC, Tsudik G, Claffy KC, Massey D, Abdelzaher T, Wang L, Crowley P, Yeh E. Named data networking (NDN) project. Technical Report, NDN-0001, Los Angeles: UCLA, 2017.

[8] Wu H, Shi JX, Wang YX, Wang YL, Zhang G, Wang Y, Liu B, Zhang BC. On incremental deployment of named data networking in local area networks. In: Proc. of the Symp. on Architectures for Networking and Communications Systems. Piscataway: IEEE Press, 2017. 82−94. [doi: 10.1109/ANCS.2017.18]

[9] Liang CC, Yu FR, Zhang X. Information-centric network function virtualization over 5G mobile wireless networks. IEEE Network, 2015,29(3):68−74. [doi: 10.1109/MNET.2015.7113228]

[10] Wang Y, He KQ, Dai HC, Meng W, Jiang JC, Liu B, Yan C. Scalable name lookup in ndn using effective name component encoding. In: Proc. of the 2012 IEEE 32nd Int'l Conf. on Distributed Computing Systems. Piscataway: IEEE, 2012. 688−697. [doi: 10.1109/ICDCS.2012.35]

[11] Wang Y, Zu Y, Zhang T, Peng KY, Dong QF, Liu B, Meng W, Dai HC, Tian X, Xu ZH, Wu H, Yang D. Wire speed name lookup: A GPU-based approach. In: Proc. of the 10th USENIX Conf. on Networked Systems Design and Implementation. Berkeley: USENIX Association, 2013. 199−212.

[12] Shi JX, Liang T, Wu H, Liu B, Zhang BC. NDN-NIC: Name-based filtering on network interface card. In: Proc. of the 3rd ACM Conf. on Information-centric Networking. New York: ACM Press, 2016. 40−49. [doi: 10.1145/2984356.2984358]

[13] So W, Narayanan A, Oran D, Wang YG. Toward fast NDN software forwarding lookup engine based on hash tables. In: Proc. of the 2012 ACM/IEEE Symp. on Architectures for Networking and Communications Systems. Piscataway: IEEE, 2012. 85−86. [doi: 10.1145/2396556.2396575]

[14] So W, Narayanan A, Oran D, Stapp M. Named data networking on a router: Forwarding at 20Gbps and beyond. In: Proc. of the ACM SIGCOMM 2013 Conf. on SIGCOMM. New York: ACM Press, 2013. 495−496. [doi: 10.1145/2486001.2491699]

[15] Le F, Xie GG, Zhang H. On route aggregation. In: Proc. of the 7th Conf. on Emerging Networking Experiments and Technologies. New York: ACM Press, 2011. 1−12. [doi: 10.1145/2079296.2079302]

[16] Degermark M, Brodnik A, Carlsson S, Pink S. Small forwarding tables for fast routing lookups. In: Proc. of the ACM SIGCOMM'97 Conf. on Applications, Technologies, Architectures, and Protocols for Computer Communication. New York: ACM Press, 1997. 3−14. [doi: 10.1145/263105.263133]

[17] Dharmapurikar S, Krishnamurthy P, Taylor DE. Longest prefix matching using Bloom filters. In: Proc. of the 2003 Conf. on Applications, Technologies, Architectures, and Protocols for Computer Communications. New York: ACM Press, 2003. 201−212. [doi: 10.1145/863955.863979]

[18] Pong F, Tzeng NF. Concise lookup tables for IPv4 and IPv6 longest prefix matching in scalable routers. IEEE Trans. on Networking, 2012,20(3):729−741. [doi: 10.1109/TNET.2011.2167158]

[19] Yi C, Afanasyev A, Wang L, Zhang BC, Zhang LX. Adaptive forwarding in named data networking. ACM SIGCOMM Computer Communication Review, 2012,42(3):62−67. [doi: 10.1145/2317307.2317319]

[20] Wang L, Hoque AKM, Yi C, Alyyan A, Zhang BC. OSPFN: An OSPF based routing protocol for named data networking. Technical Report, NDN-0003. Memphis: University of Memphis, 2012.

[21] Hoque AKMM, Amin SO, Alyyan A, Zhang BC, Zhang LX, Wang L. NLSR: Named-data link state routing protocol. In: Proc. of the 3rd ACM SIGCOMM Workshop on Information-centric Networking. New York: ACM Press, 2013. 15−20. [doi: 10.1145/2491224.2491231]

[22] Li F, Chen FY, Wu JM, Xie HY. Fast longest prefix name lookup for content-centric network forwarding. In: Proc. of the 2012 ACM/IEEE Symp. on Architectures for Networking and Communications Systems. Piscataway: IEEE, 2012. 73−74. [doi: 10.1145/2396556.2396569]

[23] Dai HC, Lu JY, Wang Y, Liu B. A two-layer intra-domain routing scheme for named data networking. In: Proc. of the 2012 IEEE Global Communications Conf. Piscataway: IEEE, 2012. 2815−2820. [doi: 10.1109/GLOCOM.2012.6503543]

[24] Garcia-Luna-Aceves JJ. A more scalable approach to content centric networking. In: Proc. of the 2015 24th Int'l Conf. on Computer Communication and Networks. Piscataway: IEEE, 2015. 1−8. [doi: 10.1109/ICCCN.2015.7288363]

[25] Yuan HW, Crowley P. Reliably scalable name prefix lookup. In: Proc. of the 2015 ACM/IEEE Symp. on Architectures for Networking and Communications Systems. Piscataway: IEEE, 2015. 111−121. [doi: 10.1109/ANCS.2015.7110125]

[26] Broder A, Mitzenmacher M. Network applications of Bloom filters: A survey. Internet Mathematics, 2004,1(4):485−509. [doi: 10.1080/15427951.2004.10129096]

[27] Fan L, Cao P, Almeida J, Broder AZ. Summary cache: A scalavble wide-area Web cache sharing protocol. ACM SIGCOMM Computer Commnication Review, 1998,28(4):254–265. [doi: 10.1145/285243.285287]

[28] Cohen S, Matias Y. Spectral Bloom filters. In: Proc. of the 2003 ACM SIGMOD Int'l Conf. on Management of Data. New York: ACM Press, 2003. 241–252. [doi: 10.1145/872757.872787]

[29] Bonomi F, Mitzenmacher M, Panigrahy R, Singh S, Varghese G. An improved construction for counting Bloom filters. In: Proc. of the 14th Conf. on Annual European Symp. London: Springer-Verlag, 2006. 684–695. [doi: 10.1007/11841036_61]

[30] Huang K, Zhang J, Zhang DF, Xie GG, Salamatian K, Liu AX, Li W. A multi-partitioning approach to building fast and accurate counting Bloom filters. In: Proc. of the 2013 IEEE 27th Int'l Symp. on Parallel and Distributed Processing. Piscataway: IEEE, 2013. 1159–1170. [doi: 10.1109/IPDPS.2013.51]

[31] Béla B. Random Graphs. 2nd ed., Cambridge: Cambridge University Press, 2001. 130–159.

[32] Afanasyev A, Moiseenko I, Zhang LX. NDNSIM: NDN simulator for ns-3. Technical Report, NDN-0005. Los Angeles: UCLA, 2012.

[33] Spring N, Mahajan R, Wetherall D, Anderson T. Measuring ISP topologies with rocketfuel. ACM SIGCOMM Computer Communication Review, 2002,32(4):133–145. [doi: 10.1145/964725.633039]

附中文参考文献:

[4] 谢高岗,张玉军,李振宇,孙毅,谢应科,李忠诚,刘韵洁.未来互联网体系结构研究综述.计算机学报,2012,35(6):1109–1119. [doi: 10.3724/SP.J.1016.2012.01109]

[5] 吴超,张尧学,周悦芝,傅晓明.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]

许志伟(1979－),男,内蒙古呼和浩特人,博士,讲师,CCF 专业会员,主要研究领域为互联网传输,可靠性和安全性优化.

陈波(1982－),男,博士,助理教授,主要研究领域为互联网传输,可靠性和安全性优化.

张玉军(1976－),男,博士,研究员,博士生导师,CCF 高级会员,主要研究领域为未来互联网体系结构.