软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2019,30(12):3765−3781 [doi: 10.13328/j.cnki.jos.005621]

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

基于概念漂移学习的 ICN 自适应缓存策略*

蔡 凌[1], 王兴伟[2], 汪晋宽[3], 黄 敏[3]

[1](东北大学 秦皇岛分校 控制工程学院,河北 秦皇岛 066004)

[2](东北大学 软件学院,辽宁 沈阳 110819)

[3](东北大学 信息科学与工程学院,辽宁 沈阳 110819)

通讯作者: 王兴伟, E-mail: wangxw@mail.neu.edu.cn

摘 要: 针对如何提高信息中心网络的网内缓存性能,提出了一种基于概念漂移学习(concept drift learning,简称CDL)的自适应缓存策略.考虑到节点数据和内容数据的相互感知对缓存性能的影响,将节点和内容的状态数据流作为网络资源,对提取的多维状态属性数据和缓存匹配数据进行分析挖掘,利用学习到的状态属性与缓存匹配之间的函数映射关系,即概念,对未来时期内的节点与内容间的匹配关系进行预测.为提高匹配算法的准确度,在学习过程中,提出了一种基于信息熵的概念漂移识别算法,当根据状态属性的信息熵变识别出漂移后,利用提出的基于概念重现的缓存算法,重新定义函数映射关系.仿真实验结果表明,该策略与 CEE,LCD,prob 和 OPP 策略相比,降低了网络运行成本,提高了用户体验质量.

关键词: 信息中心网络;缓存;数据挖掘;概念漂移;信息熵

中图法分类号: TP393

中文引用格式: 蔡凌,王兴伟,汪晋宽,黄敏.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

英文引用格式: Cai L, Wang XW, Wang JK, Huang M. Concept drift learning-based caching strategy in information-centric networks. Ruan Jian Xue Bao/Journal of Software, 2019,30(12):3765−3781 (in Chinese). http://www.jos.org.cn/1000-9825/5621.htm

Concept Drift Learning-based Caching Strategy in Information-centric Networks

CAI Ling[1], WANG Xing-Wei[2], WANG Jin-Kuan[3], HUANG Min[3]

[1](School of Control Engineering, Northeastern University at Qinhuangdao, Qinhuangdao 066004, China)

[2](College of Software, Northeastern University, Shenyang 110819, China)

[3](College of Information Science and Engineering, Northeastern University, Shenyang 110819, China)

Abstract: In order to improve the caching performance in information centric networks, an adaptive caching strategy based on concept drifting learning (CDL) was proposed. Considering the supplementary action of the node data and content data on improving caching performance, firstly, the status data flow of nodes and content were used as network resources, and then the mapping relationship, namely concept, between the multidimensional state attribution data based on the status data flow and the matching relationship value was mined. Finally, utilizing this mapping function, a matching algorithm to predict the matching relationship between the node and the content in the next time period was proposed. In order to improve the accuracy of the matching algorithm, a concept drifting detection algorithm based on information entropy was proposed. When the concept drifting of the state attribution data by the information entropy was captured, a

* 基金项目: 国家杰出青年科学基金(71325002); 辽宁省高校创新团队支持计划(LT2016007); 国家自然科学基金(61572123); 教育部-中国移动科研基金(MCM20160201); 河北省高等学校科学技术研究项目(QN2014327)

Foundation item: National Science Fund for Distinguished Young Scholars of China (71325002); Program for Liaoning Innovative Research Team in University (LT2016007); National Natural Science Foundation of China (61572123); Ministry of Education-China Mobile Research Fund (MCM20160201); Scientific Research Project of Colleges and Universities in Hebei Province (QN2014327)

收稿时间: 2017-11-13; 修改时间: 2018-02-28, 2018-05-15; 采用时间: 2018-06-25

new mapping relationship was learning by the proposed recurring concept caching algorithm. Simulation results show that CDL outperforms CEE, LCD, Prob, and OPP when looking at cost reduction of network operation and enhancement in quality of user experience.

Key words: ICN; caching; data mining; concept drifting; information entropy

互联网设计之初,用户主要的需求是联通性和资源共享.经过近 50 年的发展,以视频分发、文件下载等为代表的内容获取服务已成为互联网中的主流应用需求,用户对内容本身的关注胜于对内容位置的关注.为满足新的网络需求,提出了信息中心型网络(information centric neworking,简称 ICN)[1],将内容名字作为网络路由和传输的标识,当用户请求某一内容时,任何缓存该内容的节点都可以做出响应,从而显著提升内容传输效率.Named Data Networking(NDN)/Content-Centric Networking(CCN)[2],Data-Oriented Network Architecture (DONA)[3], Network of Information(NetInf)[4]和 ContentMediator architecture for content-aware nETworks (COMET)[5]等是典型的 ICN 类型的网络架构.

网内缓存是 ICN 的重要特征.虽然缓存理论及相关技术已经在 P2P,CDN 和 Web 等领域中得到了较为广泛和深入的研究和应用,但是它们并不适用于 ICN 架构体系,主要原因是:ICN 中的内容基于全网统一命名,具有缓存透明化的特征,可服务于不同类型的网络流量,而采用私有协议的 P2P,基于域自主命名的 Web,相同的内容很难被统一命名,难以被全网共享,且服务对象单一;ICN 的缓存具有泛在化特性,任何节点都可缓存任意内容,任何内容都可缓存在任意节点上,而且节点上缓存的内容存在被不断替换的可能,这导致缓存拓扑结构具有动态变化的特征[6−8],而 Web 和 CDN 缓存点的位置一般是确定的,缓存拓扑结构较为规则,因此需要探索和研究新的机制与策略.

适用于 ICN 的缓存机制与策略首先要解决内容部署问题.ICN 的网内缓存是作为一种基础设施服务提供给网络的,需要缓存大量内容,而路由节点的存储空间相对而言非常有限.为了充分利用有限的节点缓存资源,已经提出了一些新的缓存算法.这些算法大致可以分为 3 类:一类是基于节点数据的算法,根据节点在拓扑结构中的位置、节点的性能等,为内容选择缓存节点;一类是基于内容数据的算法,根据内容流行度选择要缓存的内容.这两类算法虽然可改善网内缓存性能,但由于节点和内容之间缺乏相互感知,前一类算法无法探知不同流行度的内容对缓存资源的不同需求,后一类算法缺乏获取网络性能的手段,不能对网内缓存资源进行有效控制.因此提出了结合节点数据和内容数据的又一类算法,根据节点和内容之间相互影响的关系,确定内容与节点的匹配关系,进行自适应的缓存.

由于ICN具有动态特性,不同时期的网络特性不同,节点状态不同,且ICN网络具有泛在缓存的特征,任何节点都可以缓存任意内容,同时,任何内容都可以缓存在任意的节点上,因此节点和内容均衍生出了海量的数据信息,而以上文献所建立的自适应数学模型与算法均未涉及对海量数据的处理与分析.已有研究表明:利用网内大数据获取的数据关联性,可更高效地分配网络资源并获得最大化的收益[9,10],这说明大数据方法可以作为分析 ICN 网络中海量的节点和内容间感知信息的技术手段,通过对不同阶段感知信息关联性的分析,可挖掘出节点和内容匹配关系的演变规律,并利用该规律进行自适应的缓存.鉴于此,本文提出将 ICN 中节点和内容的状态数据流作为网络资源,利用从中获得的节点和内容的多属性数据,挖掘出缓存决策与各属性间的依赖关系.在挖掘过程中,根据当前节点和内容的状态数据流,采用集成分类算法 Adaboost[11]对已观测到的节点和缓存内容样本进行分析,寻找其中的规律.这个规律就是当前阶段数据流所定义的概念(concept),然后利用这个规律或概念对未知节点和内容的关系进行预测.然而随着时间的推移,数据流中定义的概念也随时可能发生变化,例如,内容从发布阶段进入到流行阶段,即产生概念漂移[12],因此需要训练新的分类器以适应新到的概念.特别是,有时某些概念会在数据流中重复出现.例如:当流行期可达几个月之久的 YOUTUBE 音乐视频文件[13]和流行期相对较短的 Olympic 视频文件[14]的流行期冲突时,音乐视频流所对应的概念将发生漂移;而当 Olympic 视频文件流行期结束后,音乐视频流所对应的新概念可能会与 Olympic 视频文件发布前的概念相似或者重复.当概念重现时,本文将历史概念加入到对新样本的分类计算中,在提高缓存决策准确性的同时,降低分类器的学习代价.

本文提出的基于概念学习的自适应缓存策略,利用感知的节点数据和内容数据为驱动,实现自适应缓存的

目标,主要贡献如下.

(1) 提出大数据驱动的节点和内容的多属性表达模型,用来识别并标准化节点、内容的实时状态;

(2) 提出一种基于信息熵的概念漂移识别算法,根据节点、内容的实时状态的熵变,检测概念漂移,不需要对缓存状态进行实时标记;

(3) 提出一种基于概念重现的缓存算法,根据概念漂移结果更新分类器,重新定义多维状态属性与缓存匹配之间的函数映射关系,并在更新的过程中,考虑了概念重现的情况.

1 相关工作

网内缓存是 ICN 的重要特征,因此,如何选择合适的节点部署恰当的内容、最大化利用有限的缓存资源是研究的重点.ICN 提出之初,采用 CEE(cache everything everywhere)泛在缓存策略[15],大量冗余内容存在于网络,缓存资源浪费严重.为了提高缓存资源的利用率,一些研究提出了基于节点数据的方法.例如将命中节点的下游节点[16]或介数最大的下游节点[17]作为缓存节点.文献[18]提出一种基于网络全局节点重要度的缓存节点选择算法.文献[19]提出一种基于网络社团特性的缓存节点选择算法.文献[20]提出一种基于边缘优先逐级反馈的缓存协作策略,上游缓存节点的选择需要参考下游节点的缓存决策信息和统计信息.文献[21]提出一种综合考虑节点紧密中心性、介数中心性和度中心性等中心性指标的缓存机制.

一些研究在选择缓存节点的过程中引入了概率机制.文献[22]提出一种节点以概率 p 缓存内容的算法 Prob (copy with PROBability).在文献[23,24]中,缓存概率参考了缓存节点与源节点之间的距离以及缓存容量等因素.文献[25]中提出的缓存概率因子反比于请求者与源服务器间的距离.文献[26]提出的 MBP(max-benefit probability-based caching)策略,缓存概率因子考虑的是节点的替换代价.这些研究主要考虑节点的属性,把节点的位置和缓存容量等作为缓存选择的主要依据,内容在空间上的分布没有变得更加均匀和合理.

基于内容数据的网内缓存算法主要关注的是内容流行度的特性.文献[27]提出了一种基于流行度的缓存内容选择方法,流行度计算的是基于本地流行度进行加权求得的全局流行度.文献[28]提出了一种启发式概率缓存方法,基于缓存收益和内容热度等因素计算内容的被缓存概率.另外,还有一些研究工作关注的是内容的使用效率,例如,文献[29,30]将内容的使用效率作为是否缓存的判断依据.文献[31]提出的 StreamCache 策略从流的角度出发,将流细分为不同的内容块,通过统计不同内容块的使用效率,选择需要缓存的内容块.基于内容数据的缓存算法虽然考虑了缓存内容的选择问题,但由于缺少对网络环境、节点状态的考虑,可能造成对节点缓存资源利用的不合理问题.例如,高流行度的内容可能被缓存在高访问频率、高负载率的节点,这将导致该内容在节点上被替换的概率也较高.

为了研究内容与节点的自适应缓存问题,实现内容资源和节点缓存资源的合理配置,部分研究采用结合节点数据和内容数据的方法,在节点和内容状态相互感知的情况下进行节点和内容的匹配.对内容状态的度量主要是使用流行度,对节点状态的刻画则不尽相同,例如,文献[32,33]将缓存容量作为研究节点状态的主要参考因素,文献[34,35]是将缓存节点与源节点之间的跳数作为节点的主要特性,其中,文献[34]中的 MAGIC(max-gain in- network caching)算法考虑的是跳数的减少率,而文献[35]中的 OPP(opportunistic on-path)算法计算的是缓存节点至源节点的跳数与路径总跳数的比值,文献[36,37]中的节点状态主要是基于其在拓扑结构中的位置,文献[38]考虑的则是节点的路径跳数和缓存容量.这些算法虽然从不同角度综合考虑了内容数据与节点数据,但是它们缺乏网络整体的视角,没有从全网的角度深层次地挖掘和分析节点数据和内容数据的关联性,也没有进一步考虑不同时间阶段,即不同概念间的相互影响关系.

上述成果为基于概念学习的缓存研究提供了基础.本文所提出的缓存机制与策略,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.

2 基于概念漂移学习的缓存策略

2.1 多维状态属性数据的定义

为了实现缓存内容与网络节点的匹配,将适当的内容部署在恰当的节点上,缓存策略不仅需要考虑节点的当前状态,还需要考虑内容的当前属性.因此,缓存策略所需的数据应是能描述节点特性和内容属性等内容的多维数据.本文从数据分析的角度出发,按照以下两个维度进行数据提取[39].

(1) 在节点的维度上,计算节点的缓存率及缓存替换率;

(2) 在内容的维度上,定义内容在节点上的流行度及请求内容权重,刻画出内容热度与节点的相关性.

2.1.1 节点维度

不同位置的节点在不同的时期具有不同的访问热度,热度的变化可以通过节点缓存负载的变化来描述.节点维度就是通过缓存率和缓存替换率来分别描述节点在不同时期的负载状态.

定义节点缓存率为 CR:

$$CR=\frac{\sum_{i=0}^{n}CCS_i}{CCS(v)},$$

其中,$CCS(v)$为节点 v 的缓存容量,CCS_i 则表示该节点在单位时间内被缓存的第 i 个内容的大小.缓存率可有效地描述轻载时的缓存负载率.

定义节点缓存替换率为 RR:

$$RR=\frac{\sum_{i=0}^{n'}RCS_i}{CCS(v)},$$

其中,RCS_i 表示该节点在单位时间内被替换的第 i 个内容的大小.假设网络进入稳定状态后,大多数节点缓存空间被占满,此时,缓存替换率可以有效地描述节点的负载和缓存状态,并能反映出不同内容在节点的时效性[40].节点缓存率与节点缓存替换率的结合构成了对节点完整状态的描述.

2.1.2 内容维度

任何流行内容在时间上都会经历上升期、流行高峰到最后衰减的动态变化过程,而且内容的流行程度也受到地域位置因素的影响,同一内容在不同节点上的流行程度也不尽相同.内容维度就是分析内容流行程度与时间和空间的相关性,流行度描述了内容的流行程度随时间的动态变化趋势,本文提出的请求内容权重则是在借鉴 IDF(逆文档频率)概念[41]的基础上分析内容与节点的空间相关性.

定义节点内容的流行度为 LP_{vi}:

$$LP_{vi}=\frac{IRN_{vi}}{\sum_{i=1}^{n''}IRN_{vi}},$$

其中,IRN_{vi} 计算的是单位时间内用户在节点 v 上对内容 i 的请求量,$\sum_{i=1}^{n''}IRN_{vi}$ 统计的是用户对该节点的总请求量.

定义请求内容权重为 RW_{vi}:

$$RW_{vi}=\lg\frac{m}{m(i)},$$

其中,$m(i)$为请求内容 i 的节点数量,m 为节点总量. $\lg\frac{m}{m(i)}$ 是关于节点集合范围的全局因子,只关注节点数量,不关注具体的节点.通过分析可知,当较少的节点请求内容 i 时,权重值较大,表明节点 v 与内容 i 有着较强的相关性,因此,利用该权重值可区分出不同内容间的相对重要性.

2.2 匹配值的定义

内容与节点是否匹配可建模为二分类问题,匹配值对应的就是分类结果.

TF-IDF(词频-逆文档频率)[41]算法可以有效地评估一个词对一个语料库中一篇文档的重要程度,而缓存黏度也用于评估内容的重要性,因此在借鉴 TF-IDF 思想的基础上,定义内容的缓存黏度为 CV_{vi}:

$$CV_{vi}=CRN_{vi}\times\lg\frac{m}{m'(i)},$$

其中,$m'(i)$为缓存内容 i 的节点数量;$CRN_{vi}=\dfrac{crn_{vi}}{\sum_{i=1}^{n}crn_{vi}}$,$crn_{vi}$ 计算的是单位时间内用户对缓存在节点 v 上的内容 i 的访问量,$\sum_{i=1}^{n}crn_{vi}$ 统计的是单位时间内用户对该节点上所有缓存内容的总访问量.缓存黏度正比于用户对节点上缓存内容的访问率,而反比于缓存节点数量.通过缓存黏度的计算可以获得节点与被缓存内容的匹配程度,值越大,意味着节点 v 与缓存内容 i 的匹配度越大,缓存越适宜.

定义匹配度的门限值 TH,当缓存黏度 $CV_{vi}\geqslant TH$ 时,表明当前内容与节点具有较高的匹配度,适宜缓存,则令匹配值为 1;否则,令匹配值为−1.TH 根据第 1 次采集到的缓存黏度集合的中位值来确定.采用中位值主要是为了避免极端缓存黏度值对匹配判断的影响,并避免训练样本分布不均匀导致训练的不准确.

2.3 训练集与测试集的定义

训练集与测试集的符号描述借鉴文献[39]中的部分定义,其中,

定义属性向量 $a_{vi}=(CR_{vi},RR_{vi},LP_{vi},RW_{vi})=(a_{vi}^1,a_{vi}^2,a_{vi}^3,a_{vi}^4)$,向量元素对应节点 v 与内容 i 的各状态属性.

定义数据集 $A_{vi}=(a_{11},a_{12},\ldots,a_{vi})$为 t 时刻的标记数据集.为简化数据集的表示方式,令 $A_{vi}=X_l=(x_1,x_2,\ldots,x_l)$,其中,$x_1=a_{11}$.

定义数据集 $Y_l=(y_1,y_2,\ldots,y_l)$为类别标记集合,其中,$y_q\in\{1,-1\}$,类别标记与匹配值相对应,若标记值 $y_q=1$,则意味着该节点与内容匹配度较高,缓存;否则,不缓存.

定义测试数据集 $X_u=(x_{l+1},x_{l+2},\ldots,x_{l+u})$是时间序列 $t+\Delta t$ 上需要计算的未标记数据集,对应的类别标记集合定义为 $Y_u=(y_{l+1},y_{l+2},\ldots,y_{l+u})$.若计算结果为 $y_u=1$,则该节点上应缓存当前内容;若 $y_u=-1$,则不缓存.

2.4 基于概念漂移学习的缓存算法

ICN 网络中存在着海量的数据包在高速传输,这些数据包构成了一种典型的数据流应用.由于网络流的分布随着网络环境动态变化,因而任何内容在一定的空间区域内,其流行程度在时间维度上都会经历从上升、流行高峰到最后衰减的过程,这个过程的变化,即概念的变化,将对流量分布造成较大的影响,因此,本文提出一种面向不同概念进行自适应学习的方法来求解缓存匹配问题.

2.4.1 网络概念漂移的识别算法

针对概念漂移的识别,很多研究主要是通过分析分类的准确率来识别是否发生了概念漂移,但基于准确率的判断需要实时地对样本类别进行标记.然而标记样本需要大量的时间和资源作为代价,因此本文提出了一种基于熵值检测的无标记概念漂移识别算法.定义两个由 $m\times n$ 个基窗口组成的滑动窗口,一个记录历史数据,一个记录当前数据,通过滑动窗口机制不断比较两个窗口的熵值来分析其差异,并根据差异性来识别是否发生漂移.其中,m 是节点数量,n 是流行度为前 20%的内容的数量(由于 ICN 中内容的访问次数与其流行度满足 Zipf-Like 定律,流行度排名为前 20%的访问量约占网络总访问量的 80%,因此以排名为前 20%的流行内容为研究对象),$m\times n$ 个基窗口即是对当前网络环境下节点和内容状态属性的较完整描述.

定义在时刻 t,当前数据窗口的形式化表达为 $NW=(x_1,x_2,\ldots,x_{mn})$,如果未发生概念漂移,则该窗口中的状态属性数据将不断地被更新,其中,$mn=l$.

定义在时刻 t,历史数据窗口的形式化表达为 $OW=\{x_1^o,x_2^o,...,x_{mn}^o\}$.

定义 j 是属性向量,即窗口 x_q 中第 j 个元素,$j\in\{1,4\}$,$q\in\{1,mn\}$.

定义 b 是元素的分支.将 x_q 中每个元素的值都进行归一化处理,并将处理后仍位于[0,1]之外的数据赋值为 0 或者 1,然后将 1 分为 10 等分,比较归一化后的数值属于哪个区间,对应的区间值则为 1.例如,当前数据为 0.32,则[0.3~0.4]区间的值为 1.

定义窗口 x_q 中元素 j 在分支 b 上出现的次数为 r_{qjb}:

$$r_{qjb}=r_{(q-1)jb}+\begin{cases}1, & \text{数据属于该分支}\\ 0, & \text{数据不属于该分支}\end{cases}.$$

定义窗口 x_q 的元素 j 在分支 b 上的熵为 $H_{qjb}=-P_{qjb}\log_2(P_{qjb})$,$P_{qjb}=r_{qjb}/mn$,表示元素 j 在分支 b 上的概率.

定义窗口 x_q 的熵为

$$H_q=\frac{1}{4}\sum_{j=1}^{4}\sum_{b=0}^{9}H_{qjb}.$$

定义前 q 个窗口的熵为

$$H=\sum\nolimits_q H_q.$$

定义历史数据的熵为 H_q^o,当前数据的熵为 H_q^N,当 $\Delta H=(H_q^N-H_q^o)/q>\varepsilon$ 时,则认为检测到第 q 个窗口时发生了概念漂移.这说明在某些情况下,只需要对 ICN 中部分节点和内容的状态属性信息进行分析,就可识别出概念漂移.漂移识别门限值ε的设置是依据 Hoeffding 边界[42]来确定的.Hoeffding 边界描述如下.

随机变量 R 的 M 个独立样本的均值和真实平均值的误差不超过ε的概率为 $1-\delta$,则:

$$\varepsilon=\sqrt{\frac{R^2\ln(1/\delta)}{2M}} \tag{1}$$

Hoeffding 边界默认$\delta=10^{-7}$,二分类问题的随机变量 $R=\log_2 2=1$,文中样本数 M 为 1 600,计算得到ε的值为 0.07.

算法 1 描述了概念漂移的识别方法.算法实际上执行的是 q 次独立的运算,每次运算都围绕一个三元组 $(H_q^N,H_q^o,\Delta H)$ 展开.OW 是历史数据窗口,记录的是从检测到概念漂移后开始的 $m\times n$ 个状态属性样本.通过窗口的不断向前滑动,检测是否$\Delta H>\varepsilon$:如果是,则识别出概念漂移,用依次到来的两组数据分别更新 OW 和 NM 中的内容,然后重复整个过程.

算法 1. 概念漂移识别算法.

```
1  Initialize: t=0, slidesize=constant
2    for q=1 to m×n do
3      slide OW and NM by slidesize points, and compute ΔH
4      if ΔH>ε then
5        t=current_time
6        report concept drift at time t and update OW and NM
7      else goto step 3
8      end if
9    end for
```

2.4.2 基于集成学习的缓存算法

当 ICN 的缓存环境发生概念漂移时,根据先前概念训练的分类器对新样本空间的适用性将逐步减弱,导致分类模型的分辨能力下降.因此,需要在漂移点引入对新概念的重新学习,更新分类器.为了避免单一分类器在不同问题上泛化表现的不同,减少因分类器选择不当导致的泛化性能不佳的风险,本文采用经典的 Adaboost (adaptive boosting)集成学习算法生成缓存匹配决策的集成分类器,通过对多种分类器的集成,提高了分类结果的准确性.

集成学习的前提是:在检测到漂移点后,需更新历史数据窗口 OW 的内容,还需要对 OW 的内容进行标记,并生成训练集$\{(x_q,y_q),q=1,2,\ldots,m\times n\}$.Adaboost 算法对训练集进行迭代学习,每次迭代生成的弱分类器的分类结果都与标记进行比较,然后根据降低分类正确样本的权重,调高分类错误样本的权重的规则来更新样本的分布,并将新样本作为下一轮学习器的输入进行新的弱分类器的学习,若干个弱分类器的加权组合就是最终的集成分类器.弱分类器选用线性分类器.用集成分类器对后续时间序列上的状态属性值 $X_u=(x_{m\times n+1},x_{m\times n+2},\ldots,x_{m\times n+u})$进行

预测,即对不断更新的当前数据窗口 NM 中的数据进行预测,预测结果 $Y_u=(y_{m\times n+1},y_{m\times n+2},\ldots,y_{m\times n+u})$就是对应的缓存匹配值.

2.4.3 基于概念重现的缓存学习算法

集成分类算法可以在检测到概念漂移时对分类器进行动态调整,以适应新出现的概念,即适应当前的 ICN 网络环境.但需注意的是:有时,某些概念会在 ICN 网络运行中重复出现.当概念重现时,如果能够将相同概念的历史分类算法迁移到当前的分类算法中,这将有效地减少分类器的学习代价,提高分类的准确率.

定义历史样本特征集合为 $HS=\{hs_1,hs_2,\ldots,hs_k\}$,$hs_k$ 中存储的是当第 k 次检测到漂移点后的新的 OW 窗口中内容对应的标记.

定义历史样本特征的哈希值集合为 $HHS=\{hhs_1,hhs_2,\ldots,hhs_k\}$,$hh_k$ 记录的是对 hs 进行β次哈希计算后得到的β个最小哈希值.

定义历史概念集合为 $HC=\{hc_1,hc_2,\ldots,hc_k\}$里面存储的是每次概念漂移后由集成学习算法 Adaboost 生成的集成分类器,与 HHS 表相对应.

设漂移后根据 Adaboost 算法生成的分类器为 hc_{k+1},将其定义为主分类器 mhc,为了将从历史概念上学习到的知识迁移到当前概念中,需从 HC 中选择一个与当前概念最为相似的概念作为辅助分类器.为从 HC 中选择最相似的分类器,每次发现新概念后,都通过 minhash[43]算法对当前的 hs 进行降维,使用β个哈希函数对 hs 求哈希值,得到β个最小哈希值 hhs,并将 hs 和 hhs 分别存入历史样本特征集合 HS 和历史样本特征的哈希值集合 HHS,然后基于 LSH(局部敏感哈希)[44]算法,将已知的 hhs 中对应的哈希值进行聚类,将相似的哈希值聚集到一起,为后续的查找节约时间.计算当前概念下样本对应β的个最小哈希值,并将当前概念下生成的哈希值分别与相似集中的每一个 hs 生成的哈希值 hhs 行相似度比较,即计算最小哈希值相同元素的个数与总元素个数之间的比值τ,比值越大,越相似.选取相似度最大的 hs 对应的 hc 作为辅助分类器 ahc,则基于概念重现的分类学习算法的表达式为

$$f:y=sign(\omega_1(ahc(x))+\omega_2(mhc(x))) \tag{2}$$

其中,ω_1和ω_2分别为辅助分类器和主分类器的权重系数.若相似度大于阈值,则令$\omega_1=\omega_2=1/2$,说明是概念的重现,预测的结果将是主、辅分类器共同作用的结果.假设主分类器 mhc 的预测准确率为δ,辅助分类器 ahc 的准确率为ϑ,显然,f 的准确率是大于δ和ϑ中的最大值,因而提高了算法的准确率.若相似度小于阈值 HTH,则令$\omega_1=0$,$\omega_2=1$,这说明是一个新生成的概念,并将该概念对应的哈希值和分类器分别加入 HHS 和 HC.详细描述见算法 2.

算法 2. 概念重现的分类学习算法.

```
1  Initialize HHS, HC, OW , HTH
2  detect concept drift at time t, compute mhc by Adaboost algorithm and hhs_{k+1} by minhash algorithm;
3  for r=1 to length (HHS);
4  compute J(hhs_{k+1},hhs_r) by LSH algorithm;
5  end for
6  J'=min(J(hhs_{k+1},hhs_r));
7  if J'≥HTH,
8     ahc=HC[r], ω1=ω2=1/2
9  else ω1=0, ω2=1 and update HHS and HC
10 end if
11 compute prediction value by f=sign(ω1(ahc(x))+ω2(mhc(x)))
```

算法执行过程中,相似度阈值 HTH 由中位值计算确定.由于每次识别出漂移后都需从历史概念集合中选择一个与当前概念最为相似的概念作为辅助分类器,即计算当前概念与历史概念的相似度,因此任意两个历史概念间都对应有一个相似度值,从历史相似度中取中位值,可有效避免极端相似度的影响,而且算法简单.

2.5 缓存资源管理系统

缓存资源管理系统结构如图 1 所示:每一个路由节点通过对 Data 数据包的统计分析,可以计算出节点维度所需的原始参数,如单位时间内节点上被缓存的内容数量或被替换的内容数量;通过对 Interest 数据包的分析,可以获得内容维度计算所需的原始参数,如单位时间内对不同内容的请求次数,也可统计出对已缓存内容的请求次数.这些原始参数作为附加内容,随着 Interest 包转发给缓存资源管理服务器节点.为了便于对旧概念学习算法模型的快速调用,在资源管理服务器中定义了一张存储历史概念的表,表结构见表 1.该服务器节点通过对获得的多维数据进行挖掘、学习,并将学习后的匹配结果通过 Data 数据包转发到各个节点上,各节点根据匹配关系缓存相应的内容,实现缓存内容的差异化.

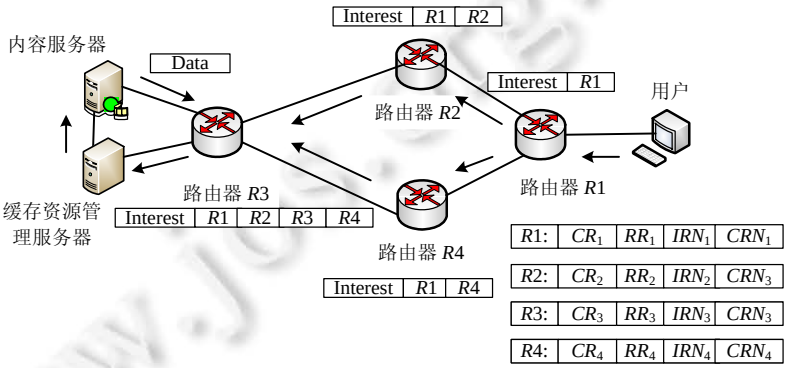


Fig.1 Resource management system architecture

图 1 缓存资源管理系统结构

Table 1 Table for storing historical concepts

表 1 存储历史概念的表结构

Id	*HHS*	*HC*
*id*1	hhs_1	hc_1
*id*2	hhs_2	hc_2
*id*3	hhs_3	hc_3
…	…	…

2.6 基于概念漂移学习的缓存策略描述

缓存策略的基本思想就是:通过对感知获得的关于节点和内容的历史数据,如缓存率、替换率、内容流行度、请求内容权重等状态特征进行分析挖掘,并利用挖掘出的状态特征与缓存匹配值之间的函数映射关系,对未来时期内的节点与内容间的匹配关系进行预测.预测方法整体流程描述如下.

首先,采用高斯方法对多维状态属性的历史数据进行归一化处理,以避免因为多维数据取值范围的不同对特征挖掘可能造成的影响.具体步骤如下.

- 首先,根据公式计算各属性值 a_{vi}^j ,$j\in(1,4)$,归一值定义为 a_{vi}^{rj} ,表达式如下:

$$a_{vi}^{rp}=0.5+\frac{a_{vi}^j-\tilde{a}_{vi}^j}{2\times 3\sigma_{vi}^j} \tag{3}$$

其中, $\tilde{a}_{vi}^j$ 是节点 v 中第 j 个属性的历史平均值, σ_{vi}^j 是节点 v 中第 j 个属性历史值的标准差.

- 然后,将归一后仍位于[0,1]之外的数据赋值为 0 或者 1,公式如下:

$$a_v^{rj}=\begin{cases}0, & a_{vi}^{rj}<0\\ 1, & a_{vi}^{rj}>0\end{cases} \tag{4}$$

在归一化处理的基础上,确定各属性值属于的分支区间.

然后,采用自适应的概念重现学习算法,利用多维状态属性值与缓存匹配值之间的函数映射关系,预测未来时期的对应关系,具体描述如算法 3 所示.

- 第 1 行、第 2 行是算法的初始化阶段;
- 第 3 行~第 9 行是当概念发生漂移后,分类算法对 *OW* 中的状态属性值所定义的概念的学习过程,包含对重现概念的迁移学习;
- 第 10 行~第 36 行是对随后到来的 *NW* 窗口中数据的处理过程,其中,
  - 第 10 行~第 17 行描述的是利用 *OW* 窗口中学习到的概念对 *NW* 窗口中数据的关系进行预测,即预测节点与内容的匹配关系;
  - 第 18 行~第 23 行描述的是与预测同时进行的概念漂移的识别过程,其中,行 18 描述的是漂移识别的算法,行 19 描述的是当识别到概念漂移后进行重新学习的过程;
  - 第 20 行~第 30 行描述的是当服务器在进行重学习时,内容与节点的匹配关系仍然按照漂移前的算法进行计算;
  - 第 31 行则说明:当重学习过程结束时,将立即采用新的学习算法预测后续内容与节点的匹配关系;
  - 第 32 行、第 33 行描述的是若未识别到概念漂移的处理动作:若未识别到概念漂移,则继续采用当前分类算法进行学习,如第 32 行所示;若对 *NW* 窗口中所有数据都已识别完成,仍未有漂移产生,则重新更新 *NW* 窗口中的数据,再进行学习和识别,如第 33 行所示.

算法 3. 自适应的概念学习算法.

Input: State attribute value $\{x_q\}_{q=1}^{mn}$ and $\{x_u\}_{mn+1}^{mn+u}$, cache viscosity $\{CV_q\}_{q=1}^{mn}$, threshold value *TH*, relationship value $\{y_q\}_{q=1}^{mn}$, entropy threshold ε, sampling numbers of *OW* T_0, sampling numbers of *NW* T_N;

Output: Relationship value $\{y_u\}_{mn+1}^{mn+u}$.

1 ω_1=0, ω_2=0, T_o=0, T_N=0;
2 *OW*=∅, *NW*=∅;
3 **when** T_o=T_o+1, update *OW*;
4 **for** *q*=1 to *m*×*n* in *OW*
5 **if** $CV_q \geqslant TH$ **then** y_q=1;
6 **else** y_q=−1;
7 **end if**
8 **end for**
9 Build classifier $f=sign(\omega_1(ahc(x))+\omega_2(mhc(x)))$ by algorithm 2
10 set T_N=T_N+1, update *NW*
11 **for** *q*=1 to *m*×*n* in *NW*
12 $Y_u=sign(\omega_1(ahc(x))+\omega_2(mhc(x)))$;
13 **if** y_u=1, **then**
14 cache the content *i* in the node *v*;
15 **else**
16 do not cache the content *i* in the node *v*;
17 **end if**
18 compute $\Delta H = H_q^N - H_q^o$ by algorithm 1;
19 **if** $\Delta H>\varepsilon$, detect concept drift, set T_o=T_o+1, goto step 3;
20 **while** (step 9 hasn't done)

```
        {
        set T_N=T_N+1, update NW
          for q=1 to m×n in NW;
            Y_u=sign(ω1(ahc(x))+ω2(mhc(x))), where Y_u is the lastest hc
              if y_u=1, then
                cache the content i in the node v;
              else
                do not cache content i in the node v;
              end if
        }
        do step 10
      else if q<mn, goto step 12;
      else if q=mn, don't detect concept drift, goto step 10;
      end if
    end for
  end
```

3 仿真实验与分析

3.1 评价指标

缓存是 ICN 网络的重要特性之一,其目的主要是为了降低网络运行成本和提高用户的体验质量,使用户能快速就近获取内容.为量化上述缓存目标,针对网络和用户分别引入不同的评价指标.在分析网络运行成本时,定义了网络链路平均利用率(所有链路单位时间利用率的平均值)、服务器负载率(网络中所有服务器单位时间接收的请求次数与用户发出的请求总次数的比值)、缓存替换数(单位时间内所有节点替换内容数量的均值)、节点负载的 Gini 系数[45](节点负载指的是节点上所有被缓存内容的总访问量,Gini 系数定义为所有节点间负载的差值之和与负载均值之比再除以节点数量平方值的 2 倍);在分析用户体验质量时,定义了访问缓存命中率(兴趣包被缓存响应的数量与用户发送的兴趣包数量的比值)、访问延时(从发出兴趣包至接收到对应的数据包所需要的时间)、访问跳数减少率(利用缓存算法获得所需内容而经过的跳数与到服务器端获取内容的跳数相比而减少的比值)、内容差异率(节点缓存的内容种类数量与网络中服务器所产生的所有内容数量的比值)等指标.在实验过程中,又分别分析了缓存容量及用户请求速率对网络运行成本及用户体验质量的影响.

在 CDL 算法性能评价方面,针对算法的准确率,定义了误报率(真实值为–1,而预测值为 1 的概率)和漏报率(真实值为 1,而预测值为–1 的概率)这两个评价指标.

3.2 实验参数设置

本文的仿真实验采用真实的域间拓扑结构 AS-1755,假设内容请求到达过程服从泊松分布,用户对内容的请求模式遵循 Zipf 分布.用户的平均请求速率为 100 个兴趣包/s,Zipf 参数设为 0.7.网络中的内容数量共有 71 000 个.每个内容缓存时需要占用一个缓存单位,网内缓存总容量范围为 0.25GB~1.5GB.实验初始时,每个节点的缓存内容量为 0.本文将 CDL 策略与 CEE 策略[15]、LCD 策略[16]、prob0.5 策略[22]和 OPP[35]策略的性能进行对比分析.针对 OPP 策略,由于其考虑了内容流行度因素但没有考虑概念漂移对流行度的影响,因此本文分析了两种场景下 OPP 策略的缓存效果:场景 1 是流行度的统计时间包含了概念漂移前与漂移后的两段时间,实验结果在图中用符号 OPPNC 表示;场景 2 是流行度的计算只基于概念漂移后的数据进行统计,实验结果用符号 OPPC 表示.

3.3 实验结果

1. 对网络运行成本的影响

- 性能指标参数随缓存容量的变化情况

图 2 为 6 种缓存策略的网络链路平均利用率随缓存容量的变化情况.可以看出,链路利用率随着缓存容量的增加而减小.这是因为随着缓存容量的增大,可缓存内容的数量也随即增多,用户可在中间缓存节点获取所需内容的概率增大,减少了缓存节点到服务器节点间链路的流量,因此链路平均利用率降低.进一步分析发现,CDL 的性能优于其他 5 种.例如:当容量为 1GB 时,CDL 与 LCD 相比,链路平均利用率降低了约 10%,与 OPPC 相比降低了约 2%.产生这一结果的原因是:CEE,LCD 和 prob0.5 在节点中缓存内容时并没有考虑节点缓存容量和转发差异性等因素,只是较盲目地进行缓存,当缓存的内容不能满足用户需求时,用户仍然需要向服务器发出请求.在这三者中,CEE 由于在节点上缓存所有内容,网络中缓存的内容量最大,用户从中间节点读取内容的概率最高,因此链路平均利用率优于另两种策略.OPPNC 与 OPPC 均考虑了内容流行度与节点位置的匹配关系,但由于在计算内容流行度的过程中,OPPNC 没有主动消弭概念漂移前的数据对流行度计算产生不良影响的手段,因此,OPPNC 对流行度的评估不如 OPPC 的准确,最终导致对链路平均利用率的优化效果不如 OPPC.而 CDL 通过不断捕捉网络概念的漂移,并从漂移点进行新概念的学习,然后利用学习的结果确定内容与节点的匹配关系,同 OPPC 策略相比,其对内容与节点的匹配关系的预测及评估更加准确,因此链路平均利用率的优化效果更佳.

图 3 为 6 种缓存策略下服务器负载率随缓存容量的变化情况.可以看出,服务器负载率均随着缓存容量的增大而减少.这是因为,随着缓存容量的增大,节点上缓存内容的总量随即增多,满足用户请求的概率相应增大,用户到服务器直接请求内容的概率相应减少.当容量为 1GB 时,CDL 与 LCD 相比,服务器负载率降低了约 10%,与 OPPC 相比降低了约 2%.

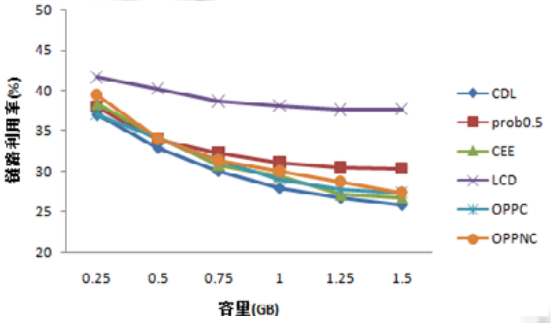


Fig.2 Average link utilization ratio

图 2 网络链路平均利用率

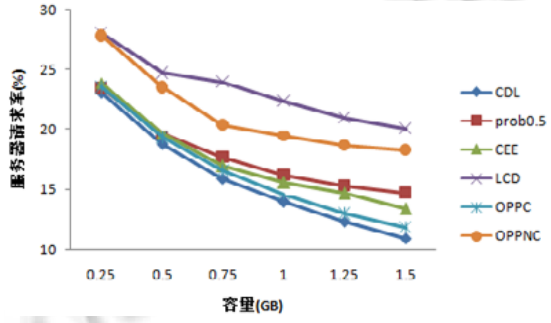


Fig.3 Server load ratio

图 3 服务器负载率

图 4 为 6 种缓存策略下缓存替换数随缓存容量的变化情况.可以看出,缓存替换数均随着缓存容量的增大而减少.进一步分析可知:CEE 策略中被替换的内容数量最多,prob0.5 策略替换数量最少.这是因为 CEE 策略的泛在缓存,节点缓存了大量的冗余内容,有效缓存容量最少,新内容的缓存将产生较多的替换操作,因此替换数量最多;prob0.5 策略基于概率缓存,缓存内容总量最少,有效缓存容量相对较多,因此当有内容缓存时,需要替换的数量最少.CDL 策略通过内容与节点的匹配计算,有效地减少了缓存替换数量,因此由缓存替换导致的节点存储操作开销也较少.

图 5 为 6 种缓存策略下节点负载 Gini 系数随缓存容量的变化情况.负载的 Gini 系数是一个用于衡量概率分布不均匀程度的测度,系数越小,说明均衡性越好.从图中可知,CDL 的 Gini 系数最小.这是因为在内容维度中定义了请求内容权重,该权重与请求节点集合密切相关;而在内容的缓存黏度的定义中也引入了缓存节点集合这一参数,因此 CDL 策略不仅只是局限于追求某个内容与节点的缓存匹配效果,还兼顾了节点集合的整体诉

求,将内容较均衡地分配给缓存节点,网络流量则被较均衡地引导到缓存节点上,降低了网络拥塞节点出现的概率,可有效减少网络运行的维护成本.

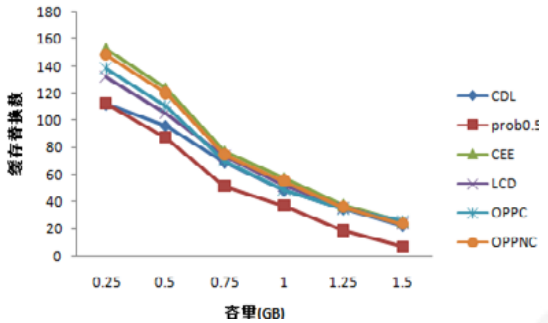


Fig.4 Cache replacement number

图 4 缓存替换数

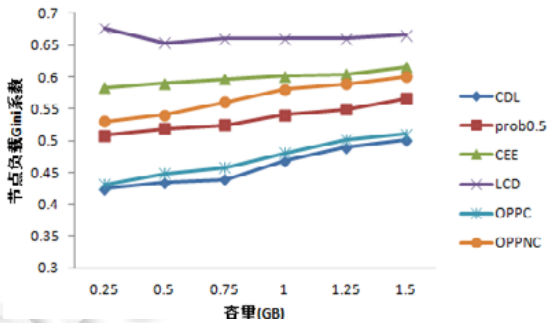


Fig.5 Node load coefficient

图 5 节点负载系数

- 性能指标参数随用户请求速率的变化情况

图 6~图 9 分别为 6 种缓存策略下网络链路平均利用率、服务器负载率、缓存替换数、节点负载 Gini 系数随用户请求速率的变化趋势.此时,缓存容量为 1GB,速率的变化范围是每个请求节点每秒发送 50~250 个请求.从图中可以看出:随着用户请求速率的增长,这 6 种策略在链路平均利用率、服务器负载率、缓存替换数、节点负载系数等方面没有明显变化,但是 CDL 策略的指标优于其余 5 种策略.

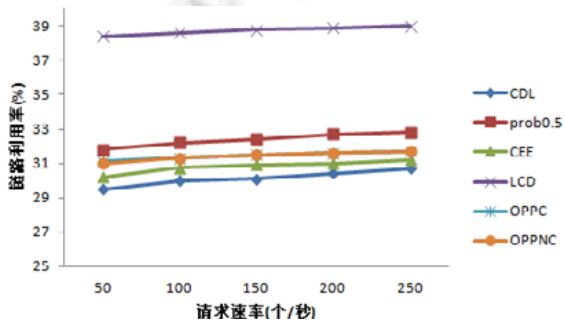


Fig.6 Average link utilization ratio

图 6 网络链路平均利用率

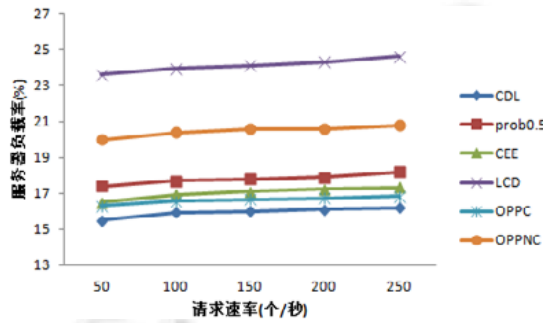


Fig.7 Server load ratio

图 7 服务器负载率

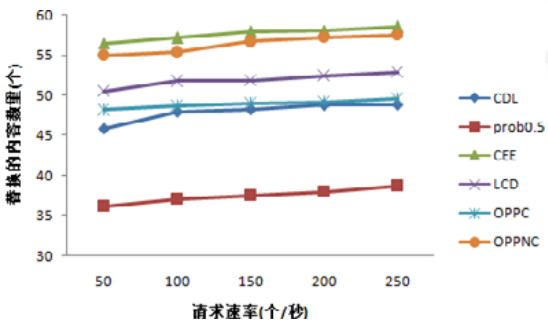


Fig.8 Cache replacement number

图 8 缓存替换数

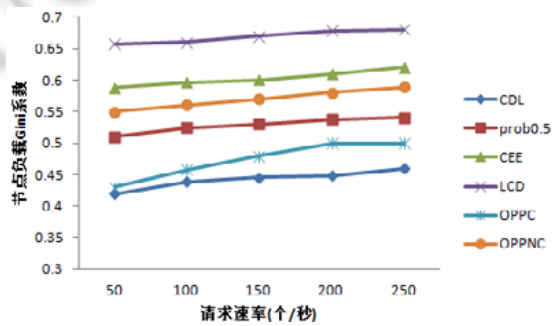


Fig.9 Node load coefficient

图 9 节点负载系数

2. 对用户体验质量的影响

- 性能指标参数随缓存容量的变化情况

图 10 为 6 种缓存策略下命中率随缓存容量的变化情况.可以看出,命中率均随着缓存容量的增大而增加.这是因为随着缓存容量的增大,中间路由器节点为用户提供的可访问内容相应增加,从而缓存命中率提升.进一步分析表明,CDL 的性能优于其他 5 种.例如:当容量为 1GB 时,CDL 与 LCD 相比,命中率提升了约 20%,与 OPP 相比提高了约 3%.产生这一结果的原因是:CEE,LCD 和 prob0.5 对内容的重复冗余缓存导致节点缓存内容频繁更新,影响了命中率;OPP 增加了流行度高的内容在靠近用户节点处的缓存概率,提高了网内缓存命中率;CDL 在时间上分析了网络的当前概念,然后在空间上把内容合理地分布在网络中不同的节点上,因而更有效地提高了网内缓存命中率.

图 11 为 6 种缓存策略下,访问跳数减少率随缓存容量的变化情况.可以看出,访问跳数减少率均随缓存容量的增大而增加.当容量为 1GB 时,CDL 与 LCD 相比,访问跳数减少率降低了约 30%,与 OPP 相比降低了约 2%.这是因为随缓存命中率的增加,用户易从更近的节点获取所需内容,访问跳数减少,从而访问跳数减少率降低.

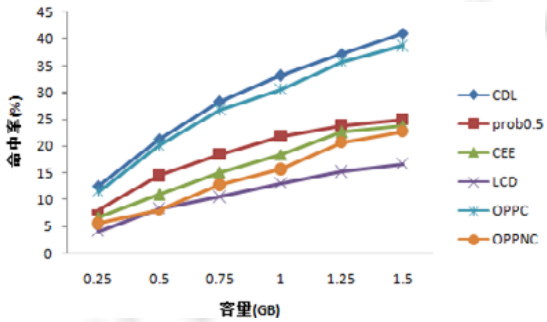


Fig.10 Cache ratio

图 10 命中率

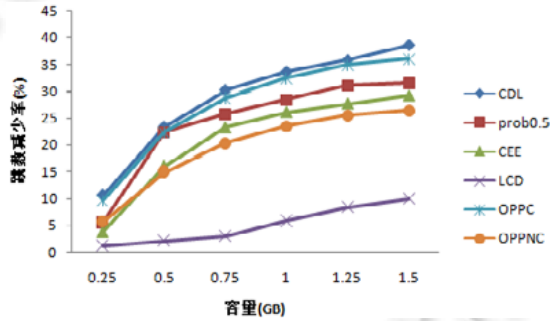


Fig.11 Hop reduction ratio

图 11 跳数减少率

图 12 为 6 种缓存策略下,访问延时随缓存容量的变化情况.随着用户获取所需内容跳数的减少,访问延时均减少.当容量为 1GB 时,CDL 与 LCD 相比,访问延时降低了约 40%,与 OPP 相比降低了约 2%.

- 性能指标参数随用户请求速率的变化情况

图 13~图 15 分别为 6 种缓存策略下命中率、跳数减少率、访问延时随用户请求速率的变化趋势.此时,缓存容量为 1GB,速率的变化范围是每个请求节点每秒发送 50~250 个请求.从图中可以看出:随着用户请求速率的增长,这 6 种策略在命中率、跳数减少率、访问延时等方面没有明显变化,但是 CDL 策略的指标优于其余 5 种策略.

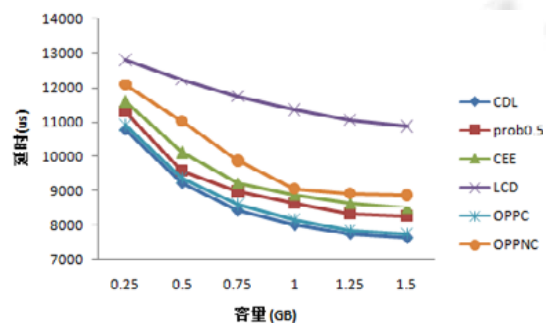


Fig.12 Delay

图 12 访问延时

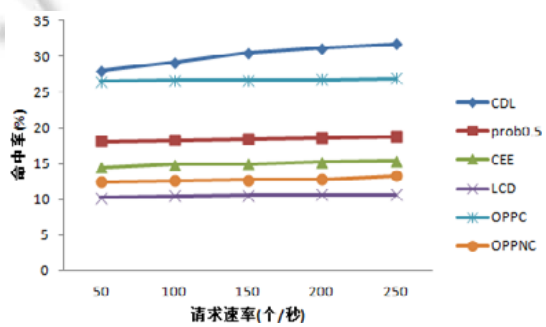


Fig.13 Cache ratio

图 13 命中率

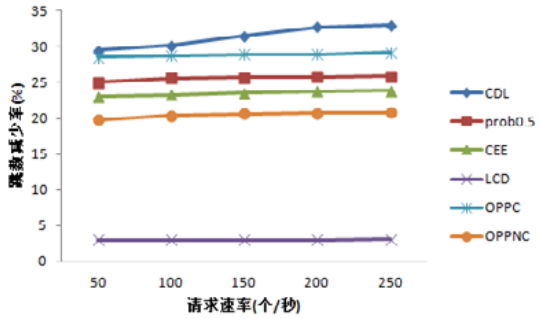


Fig.14 Hop reduction ratio

图 14 跳数减少率

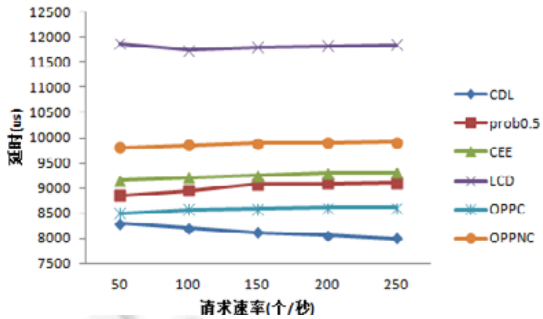
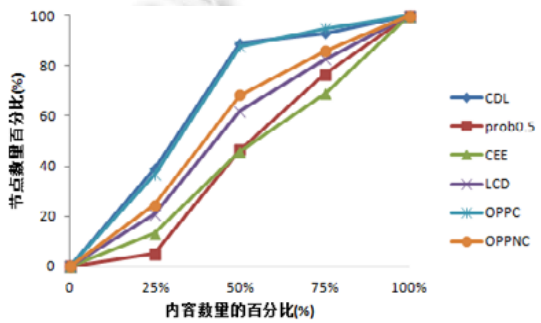


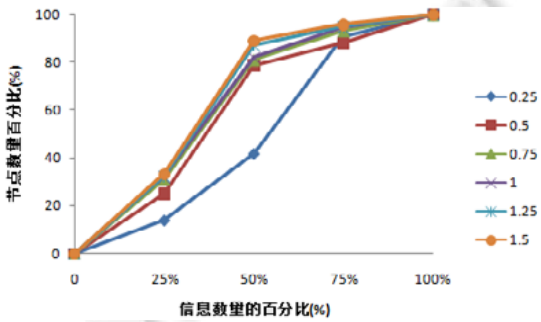
Fig.15 Delay

图 15 访问延时

图 16 是内容差异率的情况,横坐标表示的是缓存内容的类别数量与内容类别总量比值的累积分布,纵坐标表示的是节点的数量占总节点的比例.图 16(a)描述的是当缓存容量为 0.75 时,在不同的缓存策略下,内容差异率分布的变化趋势.CDL 下内容差异率小于 25%的节点数量约占 30%,差异率小于 50%的节点数约为 80%,几乎所有节点的差异率均小于 75%.图 16(b)描述的是在 CDL 下,内容差异率分布的变化趋势与缓存容量间的关系.当缓存容量大于 0.75GB 时,CDL 的内容差异率分布的变化趋势大体相同,这是因为节点能够提供足够的空间缓存匹配的内容,特定的节点缓存特定的内容,不需要其他节点的协作;而当缓存容量为 0.25GB 时,一些节点的缓存容量不足以提供满足内容存储的空间,需要缓存在其他节点,因此内容差异率小于 25%的节点数量约占 15%,差异率小于 50%的节点数约为 40%.



(a) 内容差异率随缓存策略的变化



(b) CDL 策略下内容差异率随缓存容量的变化

Fig.16 Content diversity ratio

图 16 内容差异率

3. 算法准确率分析

CDL 算法的误报率和漏报率的结果见表 2.

Table 2 False positive rate and false negative rate of CDL algorithm

表 2 CDL 算法的误报率和漏报率

容量(GB)	0.25	0.5	0.75	1	1.25	1.5
误报率(%)	18.2	13.3	17.8	16.2	17.9	18.8
漏报率(%)	13.4	19.9	20.1	20.6	24.8	24.9

从表 2 中可以看出:缓存容量的增长并未对误报率产生明显的影响,而只是漏报率产生了小幅提升.这是因为,由缓存黏度值的定义可知,黏度值与节点上被缓存内容的访问量成正比.随着缓存容量的增加,缓存替换数量减少,这就意味着节点上的某个内容在被替换前可以被多次访问,访问量增加,因此黏度值相应地增加,缓存

匹配门限值变大,因此漏报率增加;但同时,黏度值又与总访问量成反比,当某一内容的访问量增加时,总访问量也相应增加,因此黏度的增长又受到总访问量的制约,门限值的增长幅度受到限制,漏报率增长并不十分明显.

4 结束语

如何将恰当的内容部署在合适的节点上从而提高 ICN 缓存资源利用率,已经成为 ICN 缓存管理的研究重点之一.本文提出了一种基于概念漂移学习的缓存策略,在对节点和内容等多维状态属性值和缓存匹配值的历史数据学习的基础上,预测未来时期内的节点与内容间的关系.特别地,本文提出了概念漂移识别算法和基于概念重现的缓存算法,使内容能够根据当前网络环境对应的概念自适应地匹配到节点上.仿真实验结果表明:该策略不仅改善了用户体验质量,而且降低了网络运行成本.

未来研究工作的重点拟基于软件定义网络的设计思想,通过进一步感知网络实时状态信息,在控制层挖掘出节点与内容的黏度关系,确定出是否需要缓存,进而在数据层执行这一决策,从而进一步改善 ICN 性能.

References:

[1] Xylomenos G, Ververidis CN, Siris VA, Fotiou N, Tsilopoulos C, Vasilakos X, Katsaros KV, Polyzos GC. A survey of information-centric networking research. IEEE Communications Surveys & Tutorials, 2014,16(2):1024−1049.

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

[3] Koponen T, Chawla M, Chun BG, Ermolinskiy A, Kim KH, Shenker S. A data-oriented (and beyond) network architecture. ACM SIGCOMM Computer Communication Review, 2007,37(4):181−192.

[4] Dannewitz C, Golic J, Ohlman B, Ahlgren B. Secure naming for a network of information. In: Proc. of the IEEE INFOCOM. 2010. 1−6.

[5] Chai WK, Wang N, Psaras I, Pavlou G. Curling: Content-ubiquitous resolution and delivery infrastructure for next-generation services. IEEE Communication Magazine, 2011,49(3):112−120.

[6] Lü JH, Wang XW, Huang M, Shi JL, Li KQ, Li J. RISC: ICN routing mechanism incorporating SDN and community division. Computer Networks, 2017,123:88−103.

[7] Lü JH, Wang XW, Ren KX, Huang M, Li KQ. ACO-Inspired information-centric networking routing mechanism. Computer Networks, 2017,126:200−217.

[8] Lü JH, Wang XW, Huang M. Ant colony optimization-inspired ICN routing with content concentration and similarity relation. IEEE Communications Letters, 2017,21(6):1313−1316.

[9] Wolf T, Griffioen J, Calvert KL, Dutta R, Rouskas GN, Baldine I, Nagurney A. Choice as a principle in network architecture. ACM Sigcomm Computer Communication Review, 2012,42(4):105−106.

[10] Yin H, Zhang X, Zhan T, Zhang Y, Min G, Wu DO. NetClust: A framework for scalable and pareto-optimal media server placement. IEEE Trans. on Multimedia, 2013,15(8):2114−2124.

[11] Polikar R. Ensemble based systems in decision making. IEEE Circuits and Systems Magazine, 2006,6(3):21−45.

[12] Schlimmer JC, Granger RH. Incremental learning from noisy data. Machine Learning, 1986,1(3):317−354.

[13] Ahmed M, Spagna S, Huici F, Niccolini S. A peek into the future: Predicting the evolution of popularity in user generated content. In: Proc. of the ACM Int'l Conf. on Web Search and Data Mining. 2013. 607−616.

[14] Ahmed M, Traverso S, Giaccone P, Leonardi E, Niccolini S. Analyzing the performance of LRU caches under non-stationary traffic patterns. Computer Science, 2013,155(1):110−114.

[15] Ghodsi A, Shenker S, Koponen T, Raghavan B, Wilcox J. Information-centric networking: Seeing the forest for the trees. In: Proc. of the ACM HotNets. 2011. 1−6.

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

[17] Wei KC, Diliang H, Ioannis P. Cache "less for more" in information-centric networks. In: Proc. of the NETWORKING. 2012. 27−40.

[18] He Y, Zhu Y, Shi J, Zhu, N. A cache strategy in content-centric networks based on node's importance. Information Technology Journal, 2014,13(3):588−592.

[19] Cai J, Yu SZ, Liu WX. Caching strategy based on node's importance tocommunity in information-centric networks. Journal on Communications, 2015,36(6):1−10 (in Chinese with English abstract).

[20] Zhi J, LI J, Wu HB, *et al*. Edge-first-based cooperative caching strategy in information centric networking. Journal on Communications, 2017,38(3):53−64 (in Chinese with English abstract).

[21] Cai YP, Liu J, Fan XW. Node centrality metric based caching mechanism in content-centric network. Journal on Communications, 2017,38(6):10−18 (in Chinese with English abstract).

[22] Laoutaris N, Syntila S, Stavrakakis I. Meta algorithms for hierarchical Web caches. In: Proc. of the IEEE IPCCC. 2004. 445−452.

[23] Psaras I, Wei KC, Pavlou G. Probabilistic in-network caching for information-centric networks. In: Proc. of the ACM SIGCOMM ICN Workshop. 2012. 55−60.

[24] Psaras I, Wei KC, Pavlou G. In-network cache management and resource allocation for information-centric networks. IEEE Trans. on Parallel & Distributed Systems, 2014,25(11):2920−2931.

[25] Saino L, Psaras I, Pavlou G. Hashing routing scheme for information-centric networking. In: Proc. of the ACM SIGCOMM. 2013. 27−32.

[26] Wu H, Li J, Zhi J. MBP: A max-benefit probability-based caching strategy in information-centric networking. In: Proc. of the IEEE ICC. 2015. 5646−5651.

[27] Wu H, Li J, Pan T, Liu B. A novel caching scheme for the backbone of named data networking. In: Proc. of the IEEE ICC. 2013. 3634−3638.

[28] Wu HB, Li J, Zhi J. Probability-based heuristic content placement method for ICN caching. Journal on Communications, 2016, 37(5):62−72 (in Chinese with English abstract).

[29] Xu A, Tan X, Tian Y. Design and evaluation of a utility-based caching mechanism for information-centric networks. In: Proc. of the IEEE ICC. 2015. 5535−5540.

[30] Dehghan M, Massoulie L. Towsley D, *et al*. A utility optimization approach to network cache design. In: Proc. of the INFOCOM. 2016. 1−9.

[31] Li W, Oteafy SMA, Hassanein HS. StreamCache: Popularity-based caching for adaptive streaming over information-centric networks. In: Proc. of the IEEE ICC. 2016. 1−6.

[32] Liu WX, Yu SZ, Hu X, Zhu CP. Selective caching in content-centric networking. Chinese Journal of Computers, 2014,37(2): 275−288 (in Chinese with English abstract).

[33] Kim D, Lee SW, Ko YB, Kim JH. Cache capacity-aware content centric networking under flash crowds. Journal of Network & Computer Applications, 2015,50(C):101−113.

[34] Ren J, Qi W, Westphal C, Wang J. MAGIC: A distributed MAx-gain in-network caching strategy in information-centric networks. In: Proc. of the IEEE INFOCOM. 2014. 470−475.

[35] Hu X, Gong J. Opportunistic on-path caching for named data networking. IEICE Trans. on Communications, 2014,E97-B(11): 2360−2367.

[36] Wang W, Sun Y, Guo Y, Kaafar D, Jin J, Li ZC. CRCache: Exploiting the correlation between content popularity and network topology information for ICN caching. In: Proc. of the IEEE ICC. 2014. 3191−3196.

[37] Zhang G, Hu YX, Huang WW, Wang BQ, Cao LJ. Coordinated caching scheme based on popular content awareness and tracking. Journal on Communications, 2017,38(2):132−142 (in Chinese with English abstract).

[38] Li J, Feng ZM, Wu HB. Hierarchical division-based cache storage centric networking. Journal on Communications, 2016,37(1): 35−41 (in Chinese with English abstract).

[39] Cai L, Wang JK, Wang XW, Hu X. Adaptive caching algorithm based on adaboost learning for information-centric network. Journal of Northeastern University (Natural Science), 2019,40(1):24−28 (in Chinese with English abstract).

[40] Cui XD, Liu J, Huang T, Chen JY, Liu YJ. A novel in-networking caching scheme based on betweenness and replacement rate in content centric networking. Journal of Electronics & Information Technology, 2014,36(1):1−7 (in Chinese with English abstract).

[41] Wu HC, Luk RWP, Wong KF, Kwok KL. Interpreting TF-IDF term weights as making relevance decisions. ACM Trans. on Information Systems, 2008,26(3):55−59.

[42] Jin R, Agrawal G. Efficient decision tree construction on streaming data. In: Proc. of the ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining. 2003. 571−576.

[43] Broder A. On the resemblance and containment of documents. In: Proc. of the Compression and Complexityof Sequences. 1997. 21−29.

[44] Andoni A, Indyk P. Near-optimal Hashing algorithms for approximate nearest neighbor in high dimensions. In: Proc. of the IEEE FOCS. 2006. 459−468.

[45] Dixon PM, Weiner J, Mitchell-Olds T, Woodley R. Bootstrapping the Gini coefficient of inequality. Ecology, 1987,68(5): 1548−1551.

附中文参考文献:

[19] 蔡君,余顺争,刘外喜.基于节点社团重要度的 ICN 缓存策略.通信学报,2015,36(6):1−10.

[20] 智江,李俊,吴海博,等.基于边缘优先的 ICN 缓存协作策略.通信学报,2017,38(3):53−64.

[21] 蔡岳平,刘军,樊欣唯.基于节点中心性度量的内容中心网络缓存机制.通信学报,2017,38(6):10−18.

[28] 吴海博,李俊,智江.基于概率的启发式 ICN 缓存内容放置方法.通信学报,2016,37(5):62−72.

[32] 刘外喜,余顺争,胡晓,朱萍玉.CCN 中选择性缓存机制的研究.计算机学报,2014,37(2):275−288.

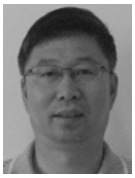
[37] 张果,胡宇翔,黄万伟,等.基于流行内容感知和跟踪的协同缓存策略.通信学报,2017,38(2):132−142.

[38] 李俊,冯宗明,吴海博,智江.基于层次划分的 CCN 网络缓存存储策略.通信学报,2016,37(1):35−41.

[39] 蔡凌,汪晋宽,王兴伟,胡曦.基于 Adaboost 学习的 ICN 自适应缓存算法.东北大学学报(自然科学版),2019,40(1):24−28.

[40] 崔现东,刘江,黄韬,陈建亚,刘韵洁.基于节点介数和替换率的内容中心网络网内缓存策略.电子与信息学报,2014,36(1):1−7.

蔡 凌(1980－),女,湖南武冈人,博士,讲师,CCF 专业会员,主要研究领域为互联网体系结构,网络路由,缓存,资源优化.



汪晋宽(1957－),男,博士,教授,博士生导师,主要研究领域为智能控制.

王兴伟(1968－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为未来网络架构,工业互联网,信息安全,云计算.

黄 敏(1968－),女,博士,教授,博士生导师,主要研究领域为物流与供应链,智能优化,生产调度.