文章编号:1007-5321(2021)02-0068-07 **DOI**:10.13190/j.jbupt.2020-086

面向新生内容需求的缓存放置与替换联合算法

单思洋, 冯春燕, 朱光宇, 张天魁
(北京邮电大学 信息与通信工程学院, 北京 100876)

摘要: 在内容中心网络中,全局缓存放置算法无法对新生内容实时缓存放置进行优化,对此,提出一种缓存放置与替换的联合优化算法,可减小缓存节点业务负载和用户内容获取的时延. 定义了整网缓存收益函数,构建了最大化整网缓存收益的最优化问题,以实现新生内容的缓存放置与已存内容的缓存替换. 为了求解所提的优化问题,将所提优化问题分解为缓存放置子问题和缓存替换子问题,提出了一种全局缓存放置优化问题的次优解,缓存放置算法的性能下限为$(1-1/e)$倍的最优解. 在缓存放置之后考虑多点协同的缓存替换算法,最小化由于缓存替换产生的缓存损失,最后通过迭代实现缓存放置与替换的联合优化. 仿真结果表明,所提联合优化算法可以提升整网缓存收益,在缓存节点负载、内容获取平均跳数和全网缓存命中率方面均优于传统方法.

关 键 词: 内容中心网络; 缓存放置算法; 缓存替换算法
中图分类号: TP393 **文献标志码**: A

A Joint Cache Placement and Replacement Algorithm for Cache Requirements of New Contents in CCN

SHAN Si-yang, FENG Chun-yan, ZHU Guang-yu, ZHANG Tian-kui
(School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China)

Abstract: In content centric networking (CCN), in order to solve the problem that the existing global cache placement algorithm cannot optimize the cache placement for new content in real time, the global cache placement and replacement algorithm for new content cache requirements is studied, reducing the cache node load and content acquisition delay for users. The network-wide cache gain function is defined, and the optimization problem of maximizing the entire network cache gain is formulated to realize the cache placement of new contents and the cache replacement of existing contents. The optimization problem is decomposed into a cache placement sub-problem and a cache replacement sub-problem. A suboptimal solution to the global cache placement optimization problem is proposed. A collaborative cache replacement algorithm to minimize the loss of cache gain due to cache replacement is proposed. Finally, the joint optimization of cache placement and replacement is realized through iterations. Simulation shows that the proposed algorithm is superior to the traditional methods in terms of cache node load, user hop count and cache hit ratio.

Key words: content centric networking; caching placement algorithm; caching replacement algorithm

收稿日期: 2020-07-07
基金项目: 国家自然科学基金项目(61971060, 61502046)
作者简介: 单思洋(1988—), 男, 博士生.
通信作者: 张天魁(1980—), 男, 教授, E-mail: zhangtiankui@bupt.edu.cn.

内容中心网络是一种新型的网络架构，摒弃以互联网协度（IP，Internet protocol）地址为路由导向的传统路由方式，改为以内容名字为地址进行路由转发，是一种面向内容分发的网络架构[1]. 网内缓存技术是内容中心网络中的关键技术之一，通过网内缓存将内容放置在网络边缘，避免重复传输大量的热点内容，可以减小网络负载，提升用户的内容获取效率. 已有研究表明，很多内容会在上线的短时间内达到请求高峰，而后热度逐渐降低，且社交网络的广泛应用进一步加速了这一过程[2]. 因此，有必要对新生内容的需求进行针对性地缓存部署，以避免造成网络拥塞. 这里的新生内容指的是刚发布的内容. 当前，缓存策略在面对新生内容时还存在较多问题. 首先，许多缓存算法为单节点缓存算法[3-5]，这类算法在进行缓存决策时只考虑了自身的需求，没有对网络整体缓存性能进行优化，缓存性能较差；其次，为了提升缓存性能，一些学者[6-8]提出了基于全局优化的缓存放置策略，这类缓存策略适用于一次性部署或者周期性部署. 当有新生内容出现时，这类全局优化的缓存策略需要整网更新缓存状态，计算复杂度很高，不能很好地实现新生内容的实时部署. 此外，在传统的缓存算法中，缓存放置算法与缓存替换算法相互独立，忽略了缓存放置和缓存替换的相互影响. 但是，面对新生内容缓存需求，新生内容的放置往往伴随着已存内容的替换，因此，缓存放置和缓存替换的联合优化将获得更好的全局优化性能.

为了实现更高的实时性和更优的缓存性能，设计了一种只针对新生内容，而非全部内容的缓存放置算法，大大降低了计算复杂度，实现新生内容的实时放置，同时结合缓存替换算法对已缓存内容进行替换，获得了更好的性能. 提出了一种缓存放置与缓存替换联合优化算法，实现了实时高效的网内缓存内容放置与替换.

1　网络模型与优化问题建模

1.1　网络模型

图 1 为网络模型的示意图. 在采用内容中心网络作为组网架构的基础上，适用的网络场景为互联网服务提供商或自制系统的管理域，域内采用的网络拓扑为任意拓扑. 在任意拓扑中，网络节点随机连接，没有遵循特定的规则.

为了获取更优的缓存性能，采用就近路由策略，优先转发至网内缓存有该内容且距离最近的缓存节点，若网内缓存中没有该内容，则将请求转发至网外内容服务器. 已有研究证明，就近路由策略能够最大化发挥缓存的性能[9-10].

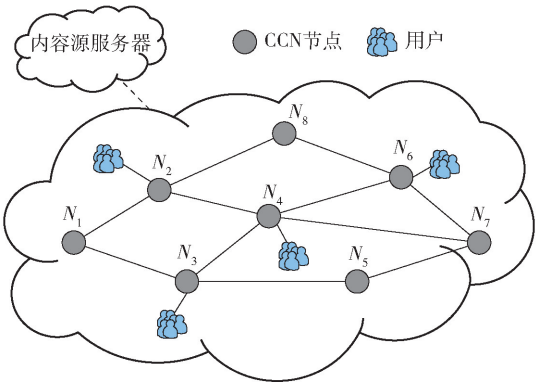

图 1　网络模型示意图

全网范围内的内容流行度分布遵循经典的 Zipf 分布[11]. 与此同时，为了体现不同节点的兴趣偏好，将各个节点处的内容请求分布进行了差异化设置. 即全网范围内的请求服从 Zipf 分布的同时，单个节点处的请求并不一定服从 Zipf 分布，具有一定的随机性，这实际上更符合现实中的场景.

1.2　优化问题建模

场景主要针对节点已缓存有内容的情况，当新生内容到达时，需要提供副本数量确定、放置位置选择以及替换内容选择的完整应对方案. 建立优化问题，目标是获得最大的整体缓存收益，这里面包含了放置新生内容带来的缓存放置收益以及替换掉已存内容造成的替换损失. 放置收益减去替换损失，即为所求的整体缓存收益.

用户 j 在采用就近路由策略情况下请求内容 i 所需的路由开销 $c_j^i(\boldsymbol{X}^i,\boldsymbol{R}^i)$ 的计算公式为

$$c_j^i(\boldsymbol{X}^i,\boldsymbol{R}^i)=\lambda_j^i\sum_{p\in\mathscr{P}_j^i}r_{j,p}^i\sum_{k=1}^{|p|-1}\omega_{p(k),p(k+1)}\prod_{k'=1}^{k}(1-x_{p(k')}) \tag{1}$$

其中：$\mathscr{P}_j^i$ 表示节点 j 请求内容 i 时的可选路径集合；$\boldsymbol{X}^i$ 为 0 或 1 放置向量，其中的元素 x_j^i 表示缓存节点 j 是否缓存有内容 i；$p(k)$ 为路径 p 中的第 k 个节点，其与缓存节点 j 的对应关系是已知的；$\boldsymbol{R}^i$ 为 0 或 1 路由矩阵，其中的元素 $r_{j,p}^i$ 为用户 j 在请求内容 i 时是否选择第 p 条路径；λ_j^i 为内容 i 在节点 j 处的请求到达率；$\omega_{p(k),p(k+1)}$ 为路径 p 中相邻节点 $p(k)$ 与节点 $p(k+1)$ 之间链路的路由开销，为定值，定义为关

于跳数的开销.

新生内容的缓存放置收益 $G(\boldsymbol{X}^{\text{new}},\boldsymbol{R}^{\text{new}})$ 和已存内容的缓存替换损失 $L(\boldsymbol{X}^{\text{ori}},\boldsymbol{Y})$ 的表达式分别定义为

$$G(\boldsymbol{X}^{\text{new}},\boldsymbol{R}^{\text{new}}) = \sum_{j\in\mathscr{J}} c_j^{\text{new}}(\boldsymbol{X}^0,\boldsymbol{R}^0) - \sum_{j\in\mathscr{J}} c_j^{\text{new}}(\boldsymbol{X}^{\text{new}},\boldsymbol{R}^{\text{new}}) \quad (2)$$

$$L(\boldsymbol{X}^{\text{ori}},\boldsymbol{Y}) = \sum_{i\in\mathscr{J}}\sum_{j\in\mathscr{J}} [c_j^i(\boldsymbol{X}^{\text{del}},\boldsymbol{R}^{\text{del}}) - c_j^i(\boldsymbol{X}^{\text{ori}},\boldsymbol{R}^{\text{ori}})] \quad (3)$$

其中:$\mathscr{J}$ 为节点集合,$\boldsymbol{X}^{\text{new}}$ 表示新生内容的放置变量,$\boldsymbol{R}^{\text{new}}$ 表示新生内容的路由变量,$\boldsymbol{Y}$ 为替换变量. 三者均为未知量. $\boldsymbol{Y}$ 为二维变量,其中的元素 $y_{i,j}$ 为1时表示将要替换掉节点 j 上缓存的内容 i,元素 $y_{i,j}$ 为0时则不替换;$\boldsymbol{X}^{\text{ori}}$ 表示网络中已存内容的放置方案,为已知量;$\boldsymbol{X}^0$ 和 $\boldsymbol{R}^0$ 分别表示网络内没有新生内容副本时的放置变量和路由变量,向量 $\boldsymbol{X}^0$ 中的元素均为0,向量 $\boldsymbol{R}^0$ 中元素的值为最短径路由策略下的默认值. $\boldsymbol{X}^{\text{ori}}$ 和 $\boldsymbol{R}^{\text{ori}}$ 分别表示网内初始的缓存放置变量和路由变量. $\boldsymbol{X}^{\text{del}}$ 是一个中间量,表示发生替换后的内容放置方案,由已知量 $\boldsymbol{X}^{\text{ori}}$ 和未知量 $\boldsymbol{Y}$ 计算得来. $\boldsymbol{X}^{\text{del}} = \boldsymbol{X}^{\text{ori}} - \boldsymbol{Y}$,表示2个矩阵中对应元素的相减操作,在确定 $\boldsymbol{X}^{\text{del}}$ 的情况下 $\boldsymbol{R}^{\text{del}}$ 的值可以通过就近路由策略得到.

定义最大化整网缓存收益 ΔG 的最优化目标为

$$\max\ \Delta G(\boldsymbol{X}^{\text{new}},\boldsymbol{R}^{\text{new}},\boldsymbol{Y}) = G(\boldsymbol{X}^{\text{new}},\boldsymbol{R}^{\text{new}}) - L(\boldsymbol{X}^{\text{ori}},\boldsymbol{Y}) \quad (4a)$$

$$\text{s.t.} \sum_{p\in\mathscr{P}_j^{\text{new}}} r_{j,p}^{\text{new}} = 1,\ \forall j \quad (4b)$$

$$\sum_{i\in\mathscr{J}} y_{i,j} = 1,\ \forall j \quad (4c)$$

$$y_{i,j} \leqslant x_{i,j},\ \forall i,j \quad (4d)$$

$$r_{j,p}^{\text{new}} \in \{0,1\},\ \forall j,p \quad (4e)$$

$$x_j^{\text{new}} \in \{0,1\},\ \forall j \quad (4f)$$

$$y_{i,j} \in \{0,1\},\ \forall i,j \quad (4g)$$

其中:$\mathscr{P}$ 为内容集合;式(4b)为新生内容的路由限制,1个节点只能选择1条路径进行路由;式(4c)为已存内容替换变量的数量限制;式(4d)为替换变量的选择限制,只能替换缓存中的已存内容;式(4e)~(4g)为变量的取值限制,均为0或1变量.

因优化模型较为复杂,最优化目标中存在3组多维变量,变量之间并不独立且相互之间存在乘积关系,难以直接求解. 为了得到性能较好的可行解,首先提出了面向新生内容的全局优化缓存放置算法,并求得特定内容副本数下的缓存放置次优解;然后提出了面向已存内容的多点协同缓存替换算法,在缓存放置次优解的基础上得出缓存替换的次优解,最终通过特定流程选出多组联合优化解中的最优值.

2 缓存放置与替换联合优化算法

基于最优化的问题,提出了缓存放置与替换的联合优化算法,通过新生内容放置算法和已存内容替换算法的交替迭代,获得整网缓存收益优化. 当有新生内容到达网内时,先利用缓存放置算法得出缓存某个数量副本时的缓存放置变量,再根据替换算法得出已存内容的缓存替换变量.

2.1 新生内容的缓存放置算法

基于式(4a)得到不考虑缓存替换损失的缓存放置子问题. 为了方便展示,省略了代表新生内容的标号 new. 新生内容在网内放置 n 个内容副本的缓存放置收益优化目标为

$$\max\ G_n(\boldsymbol{X},\boldsymbol{R}) \quad (5a)$$

$$\text{s.t.} \sum_{j\in\mathscr{J}} x_j = n \quad (5b)$$

$$\sum_{p\in\mathscr{P}_j} r_{j,p} = 1,\ \forall j \quad (5c)$$

$$r_{j,p} \in \{0,1\},\ \forall j,p \quad (5d)$$

$$x_j \in \{0,1\},\ \forall j \quad (5e)$$

为了方便,将满足限制条件的未知变量 $(\boldsymbol{X},\boldsymbol{R})$ 的集合用 $S\subset\mathbb{R}^J\times\mathbb{R}^{J\times P}$ 表示,其中 J 为节点数量,P 为可选路径数量. 与式(4a)相比,增加了内容副本总数的限制,因为在原本优化目标的缓存放置部分中,除了整数限制外没有额外的限制,只有隐含的缓存副本数不超过节点数量的限制,所以最优的解一定是副本数等于节点数的情况,缓存的内容副本越多,带来的缓存放置收益就越高.

式(5a)为NP-Hard,通过对其进行简化近似后实现求解. $c(\boldsymbol{X}^0,\boldsymbol{R}^0)$ 为网内没有新内容副本情况下的路由开销,其值为定值,因此其取值不会改变最优解. 这里,将 $\boldsymbol{R}^0$ 中的元素全部设置为1,$\boldsymbol{X}^0$ 中的元素全部为0,便于后续的近似. 由于优化问题中限制条件的表达式均为线性,且缓存放置变量和路由变量相互独立,所以可以将该问题进行松弛,将变量转化为 $\boldsymbol{\Sigma} = \{\sigma_j\}_{j\in\mathscr{J}}$ 和 $\boldsymbol{\Theta} = \{\theta_{j,p}\}_{j\in\mathscr{J},p\in\mathscr{P}}$,其取值范围均为[0,1].

首先,在 x 值为0或1的条件下,已知等式

$1-\prod_{k'=1}^{k}(1-x_{k'})=\min\left\{1,\sum_{k'=1}^{k}x_{k'}\right\}$ 成立，可以推导得到

$$G_n(\boldsymbol{X},\boldsymbol{R})=\sum_{j\in\mathscr{J}}\lambda_j\sum_{p\in\mathscr{P}_j}\sum_{k=1}^{|p|-1}\omega_{p(k),p(k+1)}\times\min\left\{1,1-r_{j,p}+\sum_{k'=1}^{k}x_{p(k')}\right\}\tag{6}$$

为了便于表示，设定函数

$$F_n(\boldsymbol{X},\boldsymbol{R})=\sum_{j\in\mathscr{J}}\lambda_j\sum_{p\in\mathscr{P}_j}\sum_{k=1}^{|p|-1}\omega_{p(k),p(k+1)}\times\min\left\{1,1-r_{j,p}+\sum_{k'=1}^{k}x_{p(k')}\right\}\tag{7}$$

根据最小值函数的凹特性 $E[\min\{x,y\}]\leqslant\min\{E[x],E[y]\}$，对 $G_n(\boldsymbol{X},\boldsymbol{R})$ 求期望，可得

$$E[G_n(\boldsymbol{X},\boldsymbol{R})]\leqslant F_n(\boldsymbol{\Sigma},\boldsymbol{\Theta})\tag{8}$$

另一方面，根据文献[11]中描述的 Goemans-Williamson 不等式，如果 $\sigma\in[0,1]$，则有

$$1-\prod_{k'=1}^{k}(1-\sigma_{k'})\geqslant\left[1-\left(1-\frac{1}{k}\right)^k\right]\min\left[1,\sum_{k'=1}^{k}\sigma_{k'}\right]\tag{9}$$

根据式(9)，将 $G_n(\boldsymbol{\Sigma},\boldsymbol{\Theta})$ 作出近似，可得

$$G_n(\boldsymbol{\Sigma},\boldsymbol{\Theta})\geqslant[1-(1-1/k)^k]\sum_{j\in\mathscr{J}}\lambda_j\sum_{p\in\mathscr{P}_j}\times\sum_{k=1}^{|p|-1}\omega_{p(k),p(k+1)}\min\left\{1,1-\theta_{j,p}+\sum_{k'=1}^{k}\sigma_{p(k')}\right\}\tag{10}$$

此外，根据不等式 $(1-1/k)^k\leqslant 1/e$，可以最终得到不等式：

$$(1-1/e)F_n(\boldsymbol{\Sigma},\boldsymbol{\Theta})\leqslant G_n(\boldsymbol{\Sigma},\boldsymbol{\Theta})\leqslant F_n(\boldsymbol{\Sigma},\boldsymbol{\Theta})\tag{11}$$

其中：$(\boldsymbol{\Sigma},\boldsymbol{\Theta})\in S'$，$S'$ 为集合 S 的凸包.

假设通过求解优化问题：

$$\max:\ F_n(\boldsymbol{\Sigma},\boldsymbol{\Theta})\tag{12}$$

满足 $(\boldsymbol{\Sigma},\boldsymbol{\Theta})\in S'$，得到的最优解为 $(\boldsymbol{\Sigma}',\boldsymbol{\Theta}')$，而后通过多项式时间内的取整算法，得到一组整数解 $(\boldsymbol{X}',\boldsymbol{R}')$，满足 $G_n(\boldsymbol{X}',\boldsymbol{R}')\geqslant G_n(\boldsymbol{\Sigma}',\boldsymbol{\Theta}')$. 取整算法的具体步骤见下文.

假设 $\max:\ G_n(\boldsymbol{X},\boldsymbol{R})$ 问题的最优解为 $(\boldsymbol{X}^*,\boldsymbol{R}^*)$，虽然其值为整数，但也包含在 S' 的范围之内. $F_n(\boldsymbol{\Sigma},\boldsymbol{\Theta})$ 的最优解为 $(\boldsymbol{\Sigma}',\boldsymbol{\Theta}')$，再根据不等式(10)，可以得到

$$G_n(\boldsymbol{X}^*,\boldsymbol{R}^*)\leqslant F_n(\boldsymbol{X}^*,\boldsymbol{R}^*)\leqslant F_n(\boldsymbol{\Sigma}',\boldsymbol{\Theta}')\leqslant\left(\frac{e}{e-1}\right)G_n(\boldsymbol{\Sigma}',\boldsymbol{\Theta}')\leqslant\left(\frac{e}{e-1}\right)G_n(\boldsymbol{X}',\boldsymbol{R}')\tag{13}$$

不等式 $G_n(\boldsymbol{X}',\boldsymbol{R}')\geqslant(1-1/e)G_n(\boldsymbol{X}^*,\boldsymbol{R}^*)$ 的意义为：通过求解得出 $G_n(\boldsymbol{X},\boldsymbol{R})$ 的可行解 $(\boldsymbol{X}',\boldsymbol{R}')$，能够保证其值的下限为最优解 $G_n(\boldsymbol{X}^*,\boldsymbol{R}^*)$ 的 $(1-1/e)$ 倍.

近似后的优化问题可简化为

$$\max\ F_n(\boldsymbol{\Sigma},\boldsymbol{\Theta})=\sum_{j\in\mathscr{J}}\lambda_j\sum_{p\in\mathscr{P}_j}\sum_{k=1}^{|p|-1}\omega_{p(k),p(k+1)}t_{j,p,k}\tag{14a}$$

$$\text{s.t.}\ \ t_{j,p,k}\leqslant 1\tag{14b}$$

$$t_{j,p,k}\leqslant 1-\theta_{j,p}+\sum_{k'=1}^{k}\sigma_{p(k')}\tag{14c}$$

$$\sum_{j\in\mathscr{J}}\sigma_j=n\tag{14d}$$

$$\sum_{p\in\mathscr{P}_j}\theta_{j,p}=1\tag{14e}$$

$$\theta_{j,p}\in[0,1]\tag{14f}$$

$$\sigma_j\in[0,1]\tag{14g}$$

在式(14a)中，共有 J 个放置变量和 PJ 个路由变量，限制条件的复杂度为 $O(PJ)$. 用辅助变量替换原目标函数中的最小值函数，可以将该问题转化为一个线性规划问题，因此，能够在多项式时间之内获得式(14a)的最优解 $(\boldsymbol{\Sigma}^*,\boldsymbol{\Theta}^*)$.

为了避免优化问题得到的最优解为分数解时无法用于实际部署，对其进行取整操作，得到整数解 $(\boldsymbol{X}',\boldsymbol{R}')$. 此外，通过取整操作，还能够保证其收益不会降低. 下面将给出取整算法的具体步骤.

首先保持路由变量 $\boldsymbol{\Theta}$ 不变，对 $\boldsymbol{\Sigma}$ 取整得到整数变量 $\boldsymbol{X}$. 任取 2 个分数值 σ_j 和 $\sigma_{j'}$，令其中一个变量变为整数(0 或 1)，所得的 $\boldsymbol{\Sigma}'$ 仍然是可行解，并且根据 ϵ-凸特性[12]，$G_n(\boldsymbol{\Sigma}',\boldsymbol{\Theta})$ 的值不会降低. 只要 $\boldsymbol{\Sigma}'$ 中依然包含分数值，就可以重复此取整操作. 这样最多可以在 $O(J)$ 个步骤中消除所有的分数变量.

对 $\boldsymbol{\Sigma}$ 取整得到整数变量 $\boldsymbol{X}$ 后，保持 $\boldsymbol{X}$ 不变，对 $\boldsymbol{\Theta}$ 进行取整，得到整数变量 $\boldsymbol{R}$. 在固定 $\boldsymbol{\Sigma}$ 的情况下，可以通过选择对每个请求节点 j 具有最大 G_n 值的路径 $p^*\in\mathscr{P}_j$ 来最大化 G_n 的值. 而这恰好是获得最低成本的路由策略，保证了每个请求节点都路由至距离最近的缓存副本处，即 $p_j^*\in\mathscr{P}_j$ 满足

$$p_j^*=\arg\min_{p\in\mathscr{P}_j}\sum_{k=1}^{|p|-1}\omega_{p_k,p_{k+1}}\prod_{k'=1}^{k}(1-x_{p_{k'}})\tag{15}$$

得出的 $\boldsymbol{R}$ 即是就近副本路由策略.

综上，可以将式(12)得到的分数解 $(\boldsymbol{\Sigma}^*,\boldsymbol{\Theta}^*)$

取整为整数解$(\boldsymbol{X}', \boldsymbol{R}')$,具体步骤为:先固定$\boldsymbol{\Theta}^*$,根据上述取整方法,将$\boldsymbol{\Sigma}^*$取整为$\boldsymbol{X}'$;之后固定$\boldsymbol{X}'$,再将$\boldsymbol{\Theta}^*$取整为$\boldsymbol{R}'$. 这样获得的整数解$(\boldsymbol{X}', \boldsymbol{R}')$,满足$G_n(\boldsymbol{X}', \boldsymbol{R}') \geqslant G_n(\boldsymbol{\Sigma}^*, \boldsymbol{\Theta}^*)$.

至此,通过松弛、近似以及求解和取整等一系列操作,得到了式(5a)的次优解$(\boldsymbol{X}', \boldsymbol{R}')$,确定了缓存放置方案和路由方案.

2.2 已存内容的缓存替换算法

现有的大部分缓存替换算法都是以单个节点为单位选择所要替换的内容,但是在就近路由策略下单节点缓存替换可能会影响整网缓存收益. 在就近路由策略下,某个节点中缓存的内容可能为多个请求节点服务. 替换掉某个节点的某个内容,可能对该节点自身的请求路由开销影响较小,但对其他节点的路由开销影响较大. 此外,还可能发生多个节点替换掉同一个内容的情况,致使网内节点对该内容的路由开销急剧上升. 因此,提出一种多节点协作的缓存替换算法,具体思路为:首先计算所有要进行替换节点中已存内容的路由开销,然后选出使路由开销损失最小的内容及其所在节点;而后在此前结果的基础上,继续选出剩余节点中使路由开销最小的内容及其节点,直至选出所有待替换节点中的内容.

算法1 多节点贪婪替换算法

输入: $\boldsymbol{X}^{\text{ori}}, \boldsymbol{R}^{\text{ori}}, \boldsymbol{\lambda}, \boldsymbol{\omega}, \boldsymbol{X}, \boldsymbol{n}$

输出: $\boldsymbol{Y}$

初始化: $y_{i,j}=0, \forall j \in \mathscr{J}, \forall i \in \mathscr{I}; \boldsymbol{X}' = \boldsymbol{X}$.

1 for $k=1:n$

2 $\quad \mathscr{J}_k = \{j: x'_j = 1, j \in \mathscr{J}\}$

3 $\quad \mathscr{I}_j^{\text{ori}} = \{i: x_{i,j}^{\text{ori}} = 1, i \in \mathscr{I}\}, j \in \mathscr{J}_k$

4 $\quad j^*, i^* = \underset{j \in \mathscr{J}_k, i \in \mathscr{I}_j^{\text{ori}}}{\arg\min} L(\boldsymbol{X}^{\text{ori}}, \boldsymbol{Y})$

where $y_{i,j}=1, y_{i',j'}=0$, for $i' \neq i, j' \neq j$

5 $\quad y_{i^*,j^*}=1$

6 $\quad x'_{j^*}=0$

7 end

在每次循环中,都将选出一个缓存节点j^*以及缓存在该节点之中的内容i^*,$\mathscr{J}_k$为等待替换的缓存节点集合,$\mathscr{I}_j^{\text{ori}}$为节点$j$中缓存的内容集合. 在选出的节点$j^*$和内容$i^*$后,将对应的替换变量元素置为1,并将与节点$j^*$相对应的$\boldsymbol{X}'$中的值置为0,以避免在下一次循环中对同一个节点进行替换. 最后循环至选出所有替换内容.

节点的缓存空间为B,第1次循环中的计算次数为BJ,第k次循环中的计算次数为$B(J-k+1)$,总的计算次数为$\frac{1}{2}BJ(J+1)$. 如果将B视作常数,则所提替换算法的计算复杂度在$O(J^2)$级别.

2.3 联合优化算法流程

由于提出的放置策略针对的是单个新生内容,在结合就近路由策略的情况下,随着网内缓存副本数的增加,带来的放置收益逐渐减小,直至收敛. 而已存内容替换策略针对的是多个内容,假设多个内容在网内的流行度水平相似,则每个内容造成的替换损失也相似,且提出的替换算法是每次都选择造成路由损失最小的内容,因此,可以近似地认为随着替换内容数量的增加,替换损失呈线性增长. 整体缓存收益为放置收益减去替换损失,因此整体缓存收益近似是一个凹函数,存在最大值. 因此,在缓存算法流程中,当整体收益开始下降时,则判断在前一个副本数处取得最大值.

当有新生内容到达网内时,缓存算法流程开始,有如下步骤. 步骤1:将新生内容副本数的限制n设置为0;步骤2:利用缓存放置算法计算得出缓存放置解$\boldsymbol{X}^n$和路由解$\boldsymbol{R}^n$;步骤3:根据缓存替换算法得出缓存替换解$\boldsymbol{Y}^n$;步骤4:利用得到的解求出整体缓存收益值ΔG_n;步骤5:将ΔG_n的值与上一个副本数的整体缓存收益值ΔG_{n-1}进行对比,若$\Delta G_n > \Delta G_{n-1}$,则进行步骤6,否则进行步骤7;步骤6:将新生内容的副本数限制增大1,而后重复步骤2;步骤7:输出前一个副本数限制下的解$\boldsymbol{X}^{n-1}$,$\boldsymbol{R}^{n-1}$和$\boldsymbol{Y}^{n-1}$,作为最终解,算法结束.

缓存放置与替换联合优化算法的循环次数在不同的模型中并不相同,但最大次数为缓存节点的总数量J,因此,联合算法的计算复杂度级别为$O(J)$.

3 仿真实现

使用的仿真平台为Python编程环境,版本为3.7. 在缓存放置的求解部分中,由于该线性规划问题的规模较大,借助了Gurobi工具箱进行辅助求解,版本为9.0.2. 缓存节点数默认为30个,采用任意拓扑结构,图2所示为仿真中采用的网内拓扑的示例. 在仿真中,内容数默认为1 000个. 单个缓存节点的缓存空间大小默认为10个,且为了方便计算,设定每个缓存空间可以容纳一个内容. 针对网络中的已存内容,Zipf分布的参数$\alpha=0.7$.

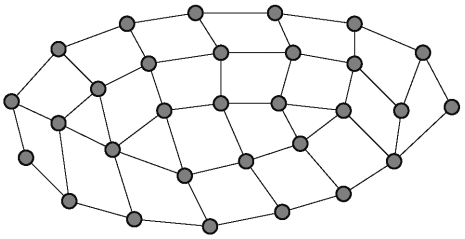

图 2　仿真拓扑示意图

图 3 所示为不同缓存放置算法下的缓存节点请求数量对比. 缓存节点服务的请求数量越高,其所承受的负载也相应的越大. 可以看出,在经典的处处缓存(LCE,leave copy everwhere)放置算法下,网络中只缓存有 3 个副本. 在采用就近路由的情况下,全网的请求都集中到这 3 个节点,给节点造成了较大的压力,也造成了局部网络链路的拥堵. 所提的缓存放置算法可以使缓存内容在节点中的分布更加均匀,大幅降低网络中负载最高的缓存节点负载. 图 1 中缓存节点 2 的负载降低约 90%.

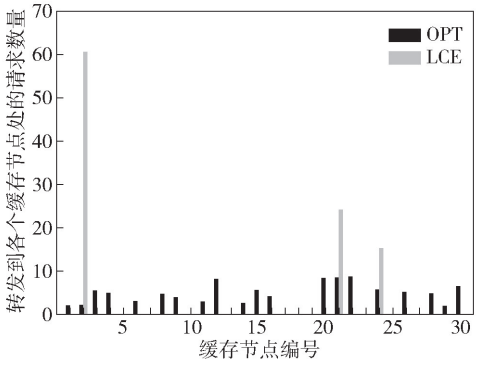

图 3　节点负载对比

图 4 和图 5 所示为在新生内容放置的不同副本数下的缓存收益性能对比. 在图 4 的场景中,新生内容在网内的流行度被设置为较高水平,而在图 5 的场景中被设置为较低水平. 首先,在缓存放置算法方面,所提缓存放置(OPL, optimization based placement)算法,在新生内容的流行度较高或较低时,缓存放置收益均优于最流行缓存放置(MPC, most popular caching)算法、随机缓存放置(RND, random caching)算法,仅次于代表最优性能的遍历缓存放置(ERG,ergodic caching)算法. ERG 算法的计算复杂度非常高,为 $O(2^J)$. 而所提 OPL 算法的复杂度为 $O(PJ)$,以大幅降低的计算复杂度获得了接近最优的缓存放置性能. 其次,在替换算法方面,由于 3 种对比算法采用的均为经典的最近最少使用替换算法(LRU,least recently used),所以其替换损失较为接近,而所提替换算法(ORP, optimization based replacement),在新生内容的流行度较高或较低时,替换损失均明显低于其他对比算法. 此外,在缓存整体收益方面,在新生内容的流行度较高或较低时,提出的算法能够获得明显高于其他对比算法的性能.

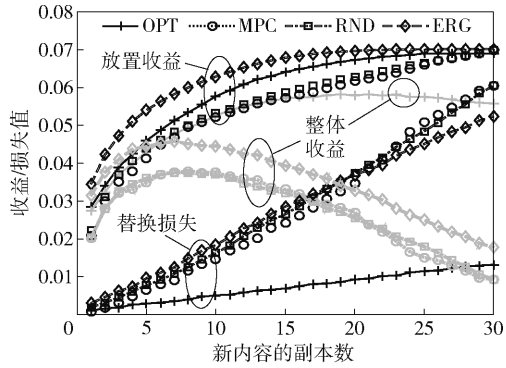

图 4　新生内容流行度水平较高时的算法性能对比

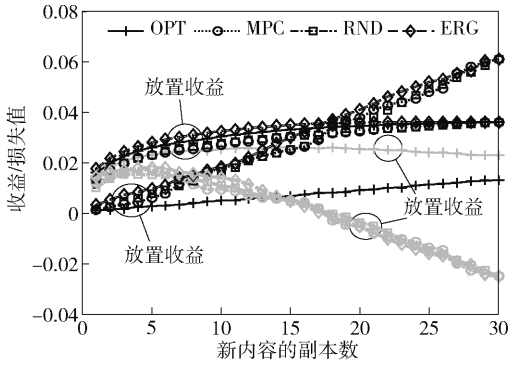

图 5　新生内容的流行度水平较低时的算法性能对比

图 6 和图 7 所示分别为不同算法的内容获取平均跳数和全网缓存命中率的性能对比. 内容获取平均跳数是指单个用户获取单个内容时所需跳数的平均值;全网缓存命中率是指网内所有请求在网内命中的比率. 仿真中的算法包括提出的缓存放置算法与替换算法的结合,用 OPL + ORP 表示;将提出的缓存放置算法与经典的 LRU 替换算法相结合,用 OPL + LRU 表示;经典的缓存放置算法 MPC,RND 和 ERG 分别表示流行度优先放置算法、随机放置算法和遍历最优放置算法,且 3 种算法均与经典的 LRU 替换算法相结合. MPC 算法与 RND 算法的平均跳数较为相近. 提出的缓存放置算法与 LRU 算法相结合的跳数性能优于 MPC 算法、RND 算法与 LRU 算法相结合的跳数性能,略差于 ERG 算法,ERG 算法比 MPC 算法跳数的最小值降低了 0.5%,

而提出的放置与替换相结合的算法在 ERG 算法的基础上,平均跳数又降低了将近 1/2. 在缓存命中率方面,所提缓存放置算法和缓存替换算法相结合下的缓存命中率性能明显高于其他算法.

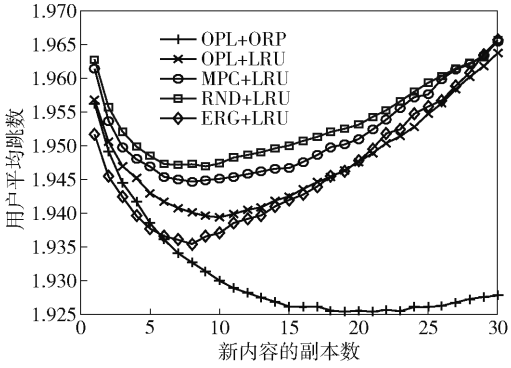

图 6 内容获取平均跳数的性能对比

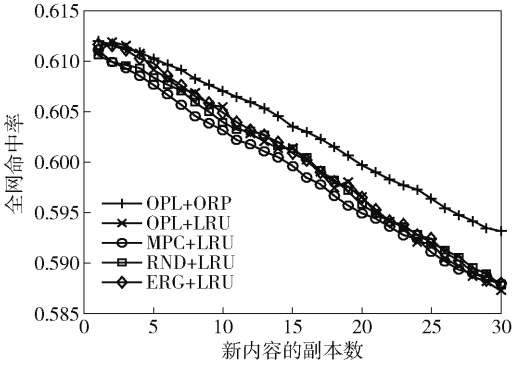

图 7 全网缓存命中率的性能对比

4 结束语

在面对网络中出现的新生内容时,现有的缓存方案存在缓存滞后,会产生网络拥塞、缓存节点负载增加、用户服务质量降低等问题. 所提的缓存放置和替换联合优化算法能够以较低的运算复杂度获得性能较优的解,缓存放置算法能够获得接近最优值的放置方案,且大幅降低由新生内容的缓存需求带来的节点负载;缓存替换算法大幅降低了替换已存内容造成的损失. 这 2 个算法降低了用户获取内容所需的平均跳数,提升了全网的缓存命中率.

参考文献:

[1] 张天魁, 单思洋, 许晓耕, 等. 信息中心网络缓存技术研究综述 [J]. 北京邮电大学学报, 2016, 39(3): 3-17.
Zhang Tiankui, Shan Siyang, Xu Xiaogeng, et al. Survey on caching techniques of information centric networking [J]. Journal of Beijing University of Posts and Telecommunications, 2016, 39(3): 3-17.

[2] Wu Jiqiang, Zhou Yipeng, Chiu Dah Ming, et al. Modeling dynamics of online video popularity [J]. IEEE Transactions on Multimedia, 2016, 18(9): 1882-1895.

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

[4] Bernardini C, Silverston T, Festor O. MPC: popularity-based caching strategy for content centric networks [C]// 2013 IEEE International Conference on Communications (ICC). Budapest: IEEE, 2013: 3619-3623.

[5] Arianfar S, Nikander P, Ott J. On content-centric router design and implications [C]// Proceedings of the Re-Architecting the Internet Workshop. New York: IEEE, 2010: 1-6.

[6] Ioannidis S, Yeh E. Jointly optimal routing and caching for arbitrary network topologies [J]. IEEE Journal on Selected Areas in Communications, 2018, 36(6): 1258-1275.

[7] Mangili M, Martignon F, Capone A. Optimal design of information centric networks [J]. Computer Networks, 2015, 91(1): 638-653.

[8] Shan Siyang, Feng Chunyan, Zhang Tiankui, et al. Proactive caching placement for arbitrary topology with multi-hop forwarding in ICN [J]. IEEE Access, 2019, 7(1): 149117-149131.

[9] Fayazbakhsh S K, Lin Y, Tootoonchian A, et al. Less pain, most of the gain: incrementally deployable ICN [C]// ACM SIGCOMM Computer Communication Review. New York: ACM, 2013: 147-158.

[10] Carofiglio G, Mekinda L, Muscariello L. Joint forwarding and caching with latency awareness in information-centric networking [J]. Computer Networks, 2016, 110(1): 133-153.

[11] Breslau L, Cao P, Fan L, et al. Web caching and Zipf-like distributions: evidence and implications [C]// INFOCOM'99. New York: IEEE, 1999: 126-134.

[12] Goemans M X, Williamson D P. New 3/4-approximation algorithms for the maximum satisfiability problem [J]. Siam Journal on Discrete Mathematics, 1994, 7(4): 656-666.