文章编号:1007-5321(2019)04-0109-05 **DOI**:10.13190/j.jbupt.2018-059

CDNs 基于缓存区预测机制的负载均衡算法

帅千钧, 王润泽, 王克勤, 金立标
(中国传媒大学 信息与通信工程学院, 北京 100024)

摘要: 提出了一种基于缓存队列长度动态预测机制的请求重定向(BMP)算法,采用 M/M/1 排队模型对分布式内容分发网络的负载均衡算法进行了描述,基于队列长度预测的机制减少了分发请求过程中不必要的重定向,同时可以克服单点过载问题. 仿真分析结果显示,相比于之前的负载均衡控制策略算法,所提出的 BMP 算法对服务器缓存区队列长度的均衡效果更优;同时,由于减少了不必要的重定向,也降低了请求响应过程的时延成本.

关 键 词: 内容分发网络; 请求重定向; 队列差异; 缓存区预测; 成本
中图分类号: TN915.5 **文献标志码**: A

Load Balancing Algorithm Based on Buffer Prediction Mechanism for CDNs

SHUAI Qian-jun, WANG Run-ze, WANG Ke-qin, JIN Li-biao
(School of Information and Communication Engineering, Communication University of China, Beijing 100024, China)

Abstract: A requests redirection algorithm for load balance based on buffer queueing length dynamic prediction (BMP) was presented. The M/M/1 model is used to describe the load balancing problem for content distribution networks. This method provides reduction of the unnecessary redirection with the queueing length prediction mechanism. And meanwhile, the overload of single point can be avoided. Evaluation shows that the proposed BMP algorithm achieved more balancing effect in server queue length compared to the so-called control law for load balancing algorithm. And the total cost of the request response time is obviously decreased due to the natively reduction of the unnecessary request redirection.

Key words: content delivery network; request redirection; queue differences; buffer prediction; cost

负载均衡算法的性能对分布式内容分发网络(CDN, content delivery network)至关重要,边缘服务器接收到请求后,可以根据自身与其他服务器的队列状态确定由本地响应还是重定向出去,从而充分利用网络资源,提高用户使用感受. 之前经典的最少连接算法、最小负载算法都是将用户请求分发到当前时间负载最轻的服务器上[1-2],但这往往造成单点过载的问题. Yang 等[3]在最少连接算法的基础上,提出了加权最少连接算法,给每台服务器赋予相应的权值,该算法在减轻负载失衡方面具有一定的作用,但每个调度周期服务器都需要交换大量的状态信息,造成不必要的通信成本. Koryachko 等[4]提出了基于服务质量保证的负载均衡算法,在服务器的选择上主要考虑了端到端的时延成本、不同链路的拥塞状况和丢包率等,但被指出不支持日益增长的网络多媒体服务,因缺乏用户体验的支持. Manfredi 等[5]针对单点过载问题,提出了请求重定向的数量与服务器之间队列长度差异成正比的负载均衡控制策略(CLB, control law for load balancing)算法,在一定程度上保证了整个系统的负载平衡. 但 CLB 算法的问题是:首先,没有考虑服务器缓存区大小的限制,认为各边缘服务器具有无限大的缓存区;其

收稿日期: 2018-03-30
基金项目: 国家科技重点研发计划项目(YFB1402203-2)
作者简介: 帅千钧(1978—), 女, 副教授, E-mail: sqj@cuc.edu.cn.

次,重定向策略让接收到请求多的服务器都向比自己接收请求少的服务器进行重定向,负载最轻的服务器仍然容易过载,造成分发的不公平性和不必要的资源浪费;再次,没有考虑请求重定向的时延成本问题. Shuai 等[6]针对 CLB 算法没有考虑服务器之间端到端时延的问题,提出了在选择重定向服务器时增加一个距离阈值的时延优化算法,避免将服务请求重定向到较远的服务器,从而有效地降低了请求响应的时延成本. 但该算法依然无法解决重定向请求不断分发到队列最短的服务器,造成某些边缘服务器缓存区出现过载的情况.

笔者在 CLB 算法的基础上,研究基于缓存队列长度预测机制的负载均衡算法.

1 问题模型和分析

分布式 CDN 一般由多个边缘服务器协同工作构成,每个服务器上均配置有一定长度的缓存用于存储服务请求队列. 具体的系统模型和服务器响应过程如图 1 所示,边缘服务器接收的服务请求通常来自于用户客户端或过载的边缘服务器. 所有分布式的服务器定期交换队列状态信息,然后各服务器的调度器将根据既定的调度策略和当前所有服务器负载情况,判断过载还是轻载,过载服务器将接收到的部分请求重定向到具有更短队列的其他服务器,而剩余的请求将缓存到本地的队列中,由本地服务器响应. 轻载的服务器要接收其他服务器重定向过来的请求,重定向的请求直接进入队列等待被响应,不会进行二次重定向,以避免多次重定向引入的时延.

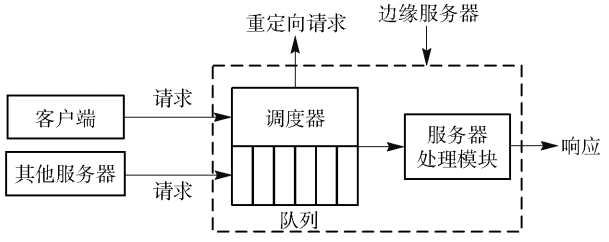


图 1 系统模型和服务器响应过程

假设各服务器每个时间周期 T 交换一次队列信息,$q_i(t)$表示服务器 i 在时刻 t 的队列长度,队列长度变化表示为

$$\Delta q_i(t) = A_i(t) + \sum_j R_{ij}(t) - C_i(t) \qquad (1)$$

其中:$A_i(t)$为一个时间周期 T 内服务器 i 接收到的来自用户的请求数量, $\sum_j R_{ij}(t)$为所有与服务器 i 有关的重定向请求数量,$C_i(t)$为服务器 i 在一个时间周期 T 内完成响应的请求数量.

服务器接收的用户请求数量与本地响应完成的请求数量可以通过统计获得,对于负载均衡算法的执行影响不大,重定向请求数则是决定负载均衡效果的关键. 根据 CLB 算法,重定向的服务请求数量与服务器之间队列长度差异成正比,定义为

$$R_{ij}(t) = \beta_{ij}(t)[q_j(t) - q_i(t)] \qquad (2)$$

其中$\beta_{ij}(t)$是正系数,用于调整重定向请求数量的大小,但不改变请求分配的比例. 由式(2)可分析得出,队列长度最小的服务器会接收最大比例数量的重定向请求. 为了后续更好的分析,可以将 $R_{ij}(t)$分成 2 个部分:一部分是由服务器 i 重定向出去的请求数;另一部分是由服务器 j 重定向到服务器 i 的请求数. $\sum_j R_{ij}(t)$即可表示为

$$\sum_j R_{ij}(t) = \sum_{j\in N_i^-(t)} R_{ij}(t) + \sum_{j\in N_i^+(t)} R_{ij}(t) \qquad (3)$$

其中:$N_i^-(t) = \{\text{server } j: q_j(t) < q_i(t)\}$, $N_i^+(t) = \{\text{server } j: q_j(t) > q_i(t)\}$. 前者代表队列长度小于服务器 i 队列长度的所有边缘服务器,后者则代表队列长度更大的所有边缘服务器. 队列长度更小,意味着更轻载,应当接收服务器 i 分发的请求;队列长度更大,负载更重,不会成为服务器 i 重定向的目标,而是向服务器 i 分发重定向请求.

为了可以准确表示服务器 i 分发出去的请求数量,系数$\beta_{ij}(t)$定义为

$$\beta_{ij}(t) = \frac{A_i(t)}{\sum_{j\in N_i^-(t)} [q_j(t) - q_i(t)]} \qquad (4)$$

由式(2)~式(4)可得

$$\sum_{j\in N_i^-(t)} R_{ij}(t) = \sum_{j\in N_i^-(t)} \frac{A_i(t)}{\sum_{j\in N_i^-(t)} [q_j(t) - q_i(t)]} \times [q_j(t) - q_i(t)] \qquad (5)$$

根据 CLB 算法,服务器 i 会将当前周期所有来自用户的请求全部按比例重定向给比自己负载更轻的边缘服务器,容易造成下个周期负载最轻的服务器出现过载的情况.

2 算法描述

针对上述问题,在前面研究的基础上,笔者提出了基于缓存区预测机制的负载均衡(BPM, load balancing algorithm based on buffer prediction mecha-

nism)算法.

调度器每间隔时间 T 执行一次调度算法,对服务器新接收到的用户请求进行分配或重定向,算法执行周期如图 2 所示.

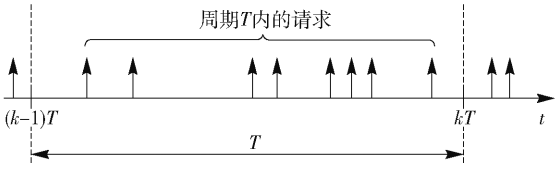


图 2　算法执行周期

在每个周期的起始阶段,服务器首先交换上一个周期结束时各服务器的状态信息,之后立即启动算法,预测本周期结束时的队列长度,据此进行重定向计算. 预测算法基于用户请求的到达模型服从均值为 λ_i 的泊松分布,服务器 i 响应请求的速度服从均值为 μ_i 的泊松分布. 例如,服务器 i 在预测周期 T 结束时,其队列长度为

$$q_i'(t+T)=q_i(t)+\lambda_i-\mu_i \tag{6}$$

令 L_{buf} 为缓存区长度,那么预测当前周期 T 结束时,过载的服务器 i 满足 $i\in\{V_i(t):q_t'(t+T)\geqslant\alpha L_{\text{buf}}\}$,$V_i(t)$ 为过载服务器的集合. 反之,轻载的服务器 j 满足 $j\in\{U_j(t):q_t'(t+T)\geqslant\alpha L_{\text{buf}}\}$,$U_j(t)$ 为轻载服务器的集合. 其中,α 是一个范围为[0,1]可调节的比例系数,用于控制轻/重载服务器的判定和确定重定向请求的数量. 轻载服务器将直接响应收到的用户请求,不再重定向请求,过载的服务器只将超出界限范围的部分请求分发给轻载的服务器. 根据前面的预测,重载服务器分发出去的请求数量设定为

$$A'_{i_{\text{redir}}}(t)=q_i(t)-\mu_i+A_i(t)-\alpha L_{\text{buf}} \tag{7}$$

由于轻载程度不同,每台轻载服务器能够接收的请求数量也是不同的. 缓存区内剩余空间越大的服务器将接收更高比例的重定向请求数量. 在确定轻载和重载服务器之后,重载服务器 i 分发给轻载服务器 j 的请求数量为

$$R_{ij}(t)=\frac{A'_{i_{\text{redir}}(t)}}{\sum_{j\in U_j(t)}[\alpha Buf-q'_j(t+T)]}[\alpha L_{\text{buf}}-q_j'(t+T)] \tag{8}$$

通过预测轻载服务器的队列长度与界限之间的差异来表示轻载程度,分发到轻载服务器的请求数量和轻载程度成正比,避免了单点过载的情况.

若 $d_{ij}(t)$ 表示服务器 i 到 j 之间的距离,定义重定向请求导致的时延成本 $\text{Cost}(t)$ 为重定向请求数量与对应重定向时延距离的乘积之和,即

$$\text{Cost}(t)=\sum_i\sum_j R_{ij}(t)d_{ij}(t) \tag{9}$$

BPM 算法执行的伪代码如下:

```
for 每一个算法执行周期 T do
  服务器之间交换各自的队列长度;
  for 服务器 1:服务器 N do
    筛选出队列长度小于自身的服务器;
    计算服务器与自身的队列长度差异,生成概率空间;
    计算接收到的请求数;
    按照概率分布分发超出的请求;
  end for
    for 服务器 1 : 服务器 N do
    处理队列中的请求,队列长度 - 处理请求速率;
    if 当前服务器为队列长度最小的服务器 then
      队列长度 = 队列长度 + 接收的请求数 + 分发到本地的请求数;
    else
      队列长度 = 队列长度 + 分发到本地的请求数;
    end if
  end for
end for
```

3　仿真分析

仿真模型采用一组由 10 个边缘服务器组成的网络拓扑结构,如图 3 所示,同时考虑了各边缘服务器之间的距离. 每个服务器都覆盖一定的用户,直接接收用户的服务请求,因此用户与直联的边缘服务器之间的距离可以忽略不计. 将每一台服务器建立具有初始队列长度和服务速率的 M/M/1 排队模型,服务器参数如表 1 所示,每个服务器连接的客户端请求数用均值为 λ 的泊松过程生成,服务器处理的请求数用均值为 μ 的泊松过程生成,算法更新的时间周期 T 设为 1 s. 仿真分析比较了所提出的 BPM 算法与 CLB 算法、时延优化算法[6]的队列均衡效果和请求响应的时延成.

服务器将无法立即响应的请求存放到缓冲区中排队等待被响应,仿真模型首先设定服务器缓存区大小为 100,为了能够找到合理的轻载和重载的界限,首先需要确定合理的 α 阈值. 图 4 是不同 α 情况下对应的服务器平均队列长度变化情况. 从图中可以看出,一开始随着α的增大,平均队列长度呈

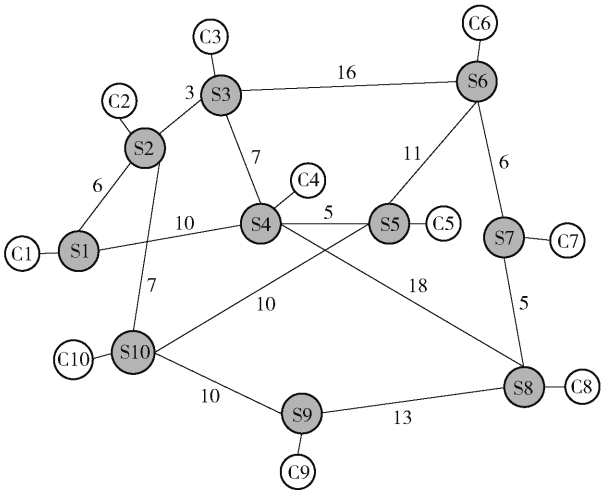


图3 仿真网络拓扑结构

表1 服务器参数

服务器	q_{i0}	λ_i	μ_i
1	20	12	12
2	10	10	13
3	5	7	14
4	25	10	17
5	20	13	10
6	10	11	11
7	5	11	11
8	25	14	7
9	10	17	10
10	20	12	12

陡降趋势;当 α 较小时,多数服务器都判为重载,轻载服务器远远少于重载服务器,容易出现单点过载,负载失衡,导致平均队列长度较大;当 α 超过18%时,随着 α 的增大,平均队列长度趋于平缓上升. 随着界限的增大,被判定为轻载的服务器数量大大增加,重载服务器数量减少,由于轻载服务器不向外分发请求,网络中重定向的请求逐渐减少,将弱化负载均衡的作用. 后续的仿真取 α 为18%来进行比较.

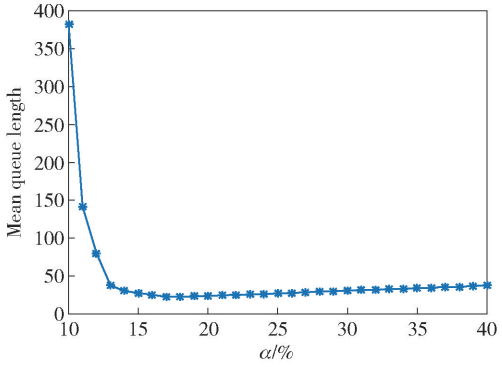
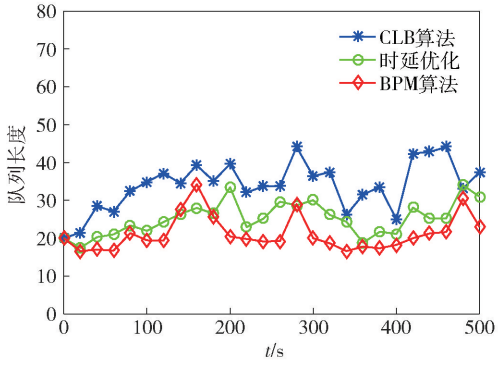
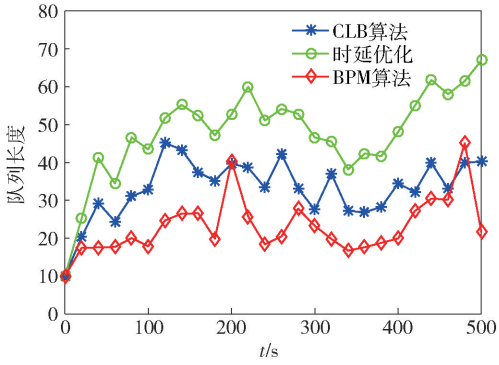


图4 不同的 α 值下服务器平均队列长度变化

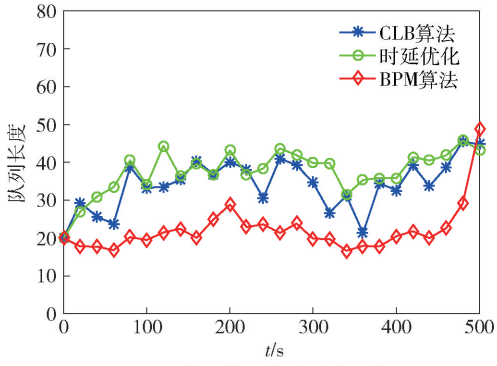
由于引入缓存区长度约束,提出的BPM算法可能无法达到与其他算法相同的性能,但能够降低分发请求的成本. 图5所示为随机选择了3台服务器的队列长度的BPM算法与CLB算法、时延优化算法的对比,以及平均队列长度的对比结果. 可以看



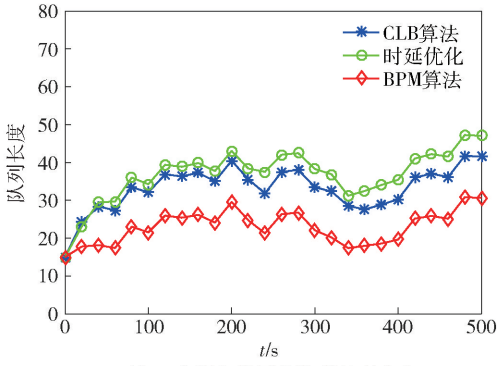
(a) 服务器1 队列长度变化



(b) 服务器6 队列长度变化



(c) 服务器10 队列长度变化



(d) 10台服务器平均队列长度变化

图5 服务器的队列长度的比较

出 BPM 算法中各服务器缓存区的队列长度更小，均衡效果更好，这是因为 BPM 算法无论是在重定向服务器选择上还是重定向请求的数量上，都选取了一个相对合理的范围，结果证实了这种改进机制能够达到更为有效的队列均衡效果.

图 6 为时延成本的对比结果，可见，BPM 算法同样减少了分发请求的数量，因而，分发请求带来的时延成本相较于 CLB 算法也得到了大幅降低.

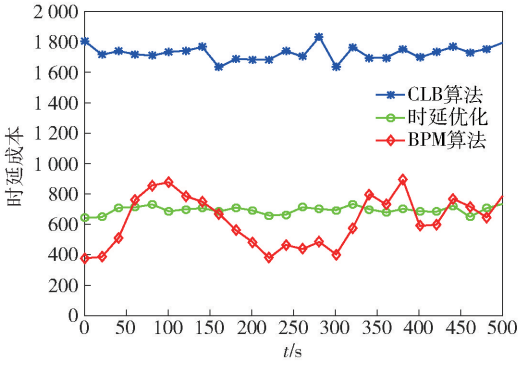


图 6　请求响应的时延成本对比

表 2 列出了详细的队列长度和时延成本的统计参数，可以看出，所提出的 BPM 算法做到了缓存队列的均衡性和请求重定向时延成本的双重兼顾，缓存队列长度下降的同时，平均成本比 CLB 算法降低了很多，而相比于时延优化算法，在时延成本不高于时延优化算法的情况下，能够取得更加有效的队列均衡效果.

表 2　算法性能参数统计

算法	队列长度均值	队列长度方差	平均时延成本
CLB	32. 300 3	1 706. 7	1 718. 4
时延优化	35. 295 2	1 932. 3	636. 8
BPM	21. 382 8	867. 1	612. 3

4　结束语

从研究结果可以看出，所提出的 BPM 算法对于分布式 CDN 能有效地优化服务成本. 与 CLB 算法相比，加入预测机制的负载均衡算法，只将超出设定界限范围的那部分请求分发出去更有针对性，能够取得更好的队列均衡效果；同时，由于减少了重定向请求的数量，请求分发过程中带来的时延成本也大幅下降，即请求响应时延得到了有效的较低，提升了服务质量和用户感受. 今后的工作将集中在进一步优化负载均衡的基础上考虑新的优化目标.

参考文献：

[1] 周松泉. 一种改进的集群动态负载均衡算法[J]. 计算机与现代化，2012(1)：135-139.
Zhou Songquan, An improved dynamic load-balancing algorithm for cluster, computer and modernization[J]. Computer and Modernization, 2012(1): 135-139.

[2] Yeung K H, Suen K W, Wong K Y. Least load dispatching algorithm for parallel web server nodes[J]. IEE Proceedings Communications, 2002, 149(4): 223-226.

[3] Yang M, Wang H, Zhao J. Research on load balancing algorithm based on the unused rate of the CPU and memory[C]//Fifth International Conference on Instrumentation and Measurement, Computer, Communication and Control. [S. l.]: IEEE, 2016: 542-545.

[4] Koryachko V, Perepelkin D, Byshov V. Approach of dynamic load balancing in software defined networks with QoS[C]//Embedded Computing. [S. l.]: IEEE, 2017: 1-5.

[5] Manfredi S, Oliviero F, Romano S P. A distributed control law for load balancing in content delivery networks [J]. IEEE/ACM Transactions on Networking, 2013, 21(1): 55-68.

[6] Shuai Q, Wang K, Miao F, et al. A cost-based distributed algorithm for load balancing in content delivery network [C] // International Conference on Intelligent Human-Machine Systems and Cybernetics. Hangzhou: IEEE, 2017: 11-15.