

基于动态反馈的一致性哈希负载均衡算法

胡丽聪, 徐雅静, 徐惠民

(北京邮电大学 信息与通信工程学院, 北京 100876)

摘 要: 负载均衡一直是分布式网络的一个研究重点. 近年来随着网络视频监控系统的广泛应用, 负载均衡也逐步应用到该领域, 以满足分布式视频监控的需要. 本文针对分布式系统的视频服务器集群提出了一种新的负载均衡方案, 将动态反馈机制与一致性哈希算法相结合, 并根据其业务特性进行了优化和改进, 增加了系统的可靠性和抗冲击性, 提高了负载均衡的总体效果.

关键词: 视频监控; 负载均衡; 动态反馈; 一致性哈希算法

中图分类号: TP36

文献标识码: A

文章编号: 1000-7180(2012)01-0177-04

A Consistent Hash Load Balancing Algorithm Based on Dynamic Feedback

HU Li-cong, XU Ya-jing, XU Hui-min

(School of Information and Communication Engineering, Beijing University of Posts and Telecomm, Beijing 100876)

Abstract: Load balancing is an important research field in distributed network. With the development of video surveillance system it is gradually applied in this field to meet the needs of distributed video monitoring in recent years. So a consistent hash load balancing algorithm based on dynamic feedback (CHDF) is proposed. It makes an optimization according to distributed video characteristics and increases the shock resistance and reliability. Simulation results show the effectiveness of algorithm.

Key words: video monitoring; load balance; dynamic feedback; consistency hash algorithm

1 引言

视频监控系统以其直观、方便、智能、可靠被广泛应用于生产管理、保安、商务等场合. 随着监控点数量的快速增长和视频质量的提高, 分布式服务器集群成为大型视频监控系统主流工作模式, 负载均衡也成为系统解决方案重要组成部分^[1]. 负载均衡算法的选择由不同的网络应用环境和业务需求决定. 目前视频监控中主要采用的负载均衡算法有随机算法, 轮询算法, 权重轮询调度算法, 最小连接调度和动态反馈调度等^[2-4]. 其中, 轮询算法将客户端请求依次分配给各个服务器, 不能很好的应用于服务器配置不同的情况; 权重轮询调度算法, 需要手动设定服务器权重, 而且不能随系统设备老化与变更

自动更新; 最小链接调度和动态反馈调度都是基于自动定时查询的, 查询间隔过大不能很好的实现负载均衡, 过小则占用较多的系统资源.

本文提出一种基于动态反馈机制的一致性哈希负载均衡算法(CHDF). 将多用户同一请求的业务特点纳入负载均衡范围, 采用动态反馈机制^[5], 结合一致性哈希算法^[6]实时调整服务器的负载, 同时处理服务器增减等突发事件, 实现了系统均衡、稳定、热变更的目标.

2 系统模型

分布式视频监控系统的逻辑架构如图 1 所示. 中央服务器接收到客户端的视频请求后, 将任务分配给相应的视频服务器节点, 然后由视频服务器节

点把实时视频源的视频流转发给客户端。

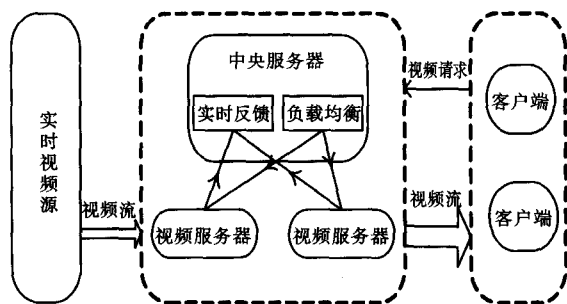


图1 系统逻辑架构图

本文的算法就是针对中央服务器如何均衡分配视频转发任务而提出的。

3 算法描述

3.1 系统工作机制

本文针对大型视频监控系统的网络波动情况,比如节点加入、退出、业务请求频率、系统资源状况、网络传输带宽限制等做了统计分析,提出了包括负载排名、负载分配、负载调整、负载转移等基于动态反馈机制的一致性哈希负载均衡算法。

负载排名算法根据动态反馈得到的每个服务器负载情况,确定是否调整管理节点负载;负载分配算法负责将系统中所有视频资源分配到各服务器,保存分配结果。负载调整算法在系统偏离均衡状态时,采取何种策略进行负载调整;负载转移算法用于系统添加或删除服务器时快速的恢复系统均衡。

为方便算法描述,本文定义如下概念:

(1) 实时负载率 R : 服务器实时的负载与满负荷运转的比例。

(2) 服务器资源数 a : 服务器的 CPU, 缓存, 网络带宽之中任何一个参数在满负荷状态下的容量。

(3) 均衡阈值 R_i : 设定服务器实时负载率之差 ($R_{\text{MAX}} - R_{\text{MIN}}$) 的上限值。

3.2 服务器节点负载排名算法:

视频服务器节点所能承受的负载主要受限于 CPU 处理能力, 缓存大小以及网络带宽等。相关变量定义如下:

C_p : 视频服务器节点当前 CPU 使用率。

C_a : 视频服务器节点当前内存使用率。

N_e : 视频服务器节点当前网络使用率。

定义 C_{p_M} 、 C_{a_M} 和 N_{e_M} 分别为 CPU 处理能力, 缓存大小以及网络带宽的阈值, 即超过阈值时, 系统的服务质量将得不到保证。定义 C_{p_N} 、 C_{a_N}

N_e 、 N_{e_N} 为系统空闲时的阈值。则编号为 i 的服务器的实时负载率为:

$$R_i = \max \left\{ \begin{aligned} & (C_p(i) - C_{p_N}(i)) / (C_{p_M}(i) - C_{p_N}(i)), \\ & (C_a(i) - C_{a_N}(i)) / (C_{a_M}(i) - C_{a_N}(i)), \\ & (N_e(i) - N_{e_N}(i)) / (N_{e_M}(i) - N_{e_N}(i)) \end{aligned} \right\}$$

中央服务器依据 R_i 的值对视频服务器节点进行负载排名。

3.3 加权一致性哈希的负载分配

加权一致性哈希的负载分配算法首先需要定义虚拟节点数量, 而且想要取得比较好的负载均衡效果, 在服务器数量比较少的时候需要增加虚拟节点。设实际服务器数目为 N , 则虚拟节点总数为 $Q = N \cdot \Omega(\log N)^{[7]}$ 。每个服务器分配到的虚拟节点数目 $Q(i) = \frac{a_i}{A} \cdot Q$ 。其中, a_i 代表编号为 i 的服务器资源数, $A = \sum a_i$ 。

然后, 将虚拟节点和视频索引进行映射。将虚拟节点和视频分别进行编号, 经过哈希函数 $H(x)$ 将虚拟节点和视频散布在 $0 \sim 2^{32}$ 的环上, $H(x)$ 为 MD5 算法, 该算法产生的值域为 $(0 \sim 2^{32} - 1)^{[7]}$ 。如图 2 所示。按顺时针方向, 两个虚拟节点之间的视频分配给第一个虚拟节点所属服务器, 以此计算虚拟节点的视频任务。

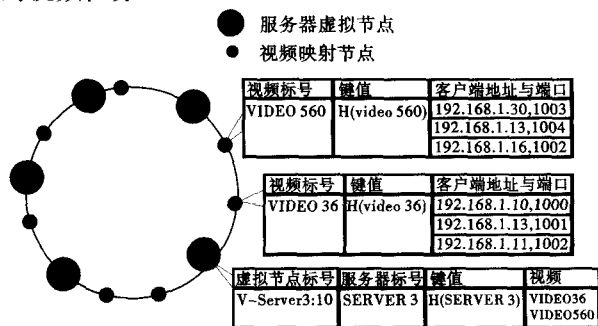


图2 服务器虚拟节点和视频资源映射图

3.4 加权一致性哈希的负载调整

在系统运行期间, 每个视频服务器上的负载都是动态变化的。中央服务器每隔时间 t 动态获取每个视频服务器的当前负载情况。设 H 为负载率最大的视频服务器标号, Z 为负载率最小的视频服务器标号, 若连续两次得到负载率之差的 ($R_H - R_Z$) 大于 R_i 时, 中央服务器启动负载调整算法。

设系统所有虚拟节点平均负载率为 R_{AVG} 。

$$R_{\text{AVG}} = \frac{\sum_{i=1}^{i=N} [R(i) \cdot a_i]}{\sum_{i=1}^{i=N} a_i}$$

$\text{Rank}(V_{H1}, \dots, V_{Hm})$ 是虚拟节点负载排序函数, 将分配给服务器 H 的虚拟节点 $\{V_{H1}, \dots, V_{Hm}\}$ 的负载 $\{L_{VH1}, \dots, L_{VHm}\}$ 降序排列, 然后 $\text{Transmit}(V_{Hj}, Z)$ 函数将服务器 H 的最小负载虚拟节点 V_{Hj} 转移至服务器 Z . 负载调整算法伪代码如下:

1. while $((R_H - R_Z) > \alpha \cdot R_i)$; // 将服务器 H 的虚拟节点按负载大小降序排列
2. $\text{Rank}(V_{H1}, \dots, V_{Hm})$;
3. $\Delta L_{\text{total}} = \min\{((R_{\text{AVG}} - R_Z) \cdot a_z), ((R_H - R_{\text{AVG}}) \cdot a_H)\}$;
4. $\Delta L = 0$;
5. while $((\Delta L < \Delta L_{\text{total}}) \&\& (R_H > R_Z))$
6. $\text{Transmit}(V_{Hj}, Z)$; // 负载转移函数
7. $j++$; // 高负载虚拟节点依次转移
8. $\Delta L += L_{VHj}$;
9. $R_H = R_{\text{refresh}}(H)$; // 服务器 H 负载率更新.
10. $R_Z = R_{\text{refresh}}(Z)$; // 服务器 Z 负载率更新.
11. end while
12. $R_{\text{refresh}}(N)$; // 更新系统所有服务器负载率.
13. $R_H = \max\{R(i) \mid i=1, \dots, N\}$; // 重新确定 H 和 L 服务
14. $R_Z = \min\{R(i) \mid i=1, \dots, N\}$; // 服务器编号, 获取负载率.
15. end while

算法执行过程中 H 和 Z 会随着内层循环执行完而变化. 通过反复运行 $\text{Transmit}(V_{Hj}, Z)$ 和更新 H, Z , 系统的实时负载率之差 $(R_H - R_Z)$ 最终缩小到 $(0 \sim \alpha \cdot R_i)$ 范围内, 这时可确定算法达到了负载调整的目标.

3.5 加权一致性哈希的负载转移

当系统增加新的服务器时, 中央服务器会根据新加入服务器的资源情况分配 ΔQ 个虚拟节点:

$$\Delta Q = \frac{a_{N+1}}{A} \cdot Q$$

新的虚拟节点经过 $H(X)$ 函数哈希映射到环上, 由于系统所有节点之间只存在 $O(\lg(N+1))$ 的哈希不平衡因子的差异, 各服务器的虚拟节点数与其资源数仍然保持线性关系^[6].

同样, 系统检测到某服务器节点异常时, 将其对应的虚拟节点的视频任务转移至相邻虚拟节点, 并删除故障服务器的虚拟节点, 保证服务器节点分得与其资源基本匹配的额外负载.

4 系统仿真与算法性能分析

系统仿真使用了 15 台 Dell R710 服务器构成视频转发服务器集群, 服务器配置为 2.27GHz

CPU, 8GB 内存, 1G 网络适配器. 通过设定不同的负载阈值模拟实现服务器配置不同的情况. 仿真参数如表 1 所示.

表 1 仿真实验的配置参数

参数	值
实时视频总数	6000 路
虚拟服务器节点总数	3000
反馈时间间隔	10s
R_i	0.1
α	0.4

本文提出的 CHDF 算法与传统的 DBA 算法 (Dynamic Biasing Algorithm)^[8] 在网络动荡、系统变更以及资源利用三方面进行了比较. 仿真结果如图 3、图 4 和图 5 所示.

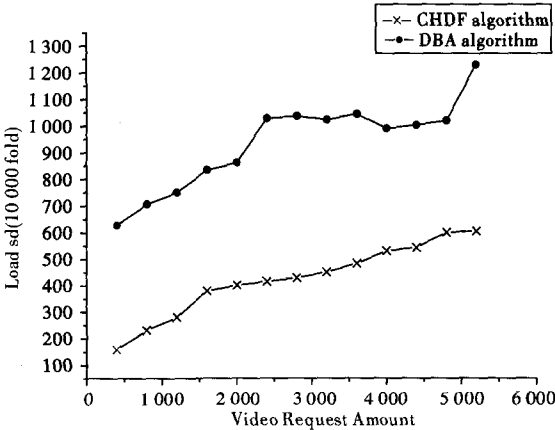


图 3 压力-稳定性对比

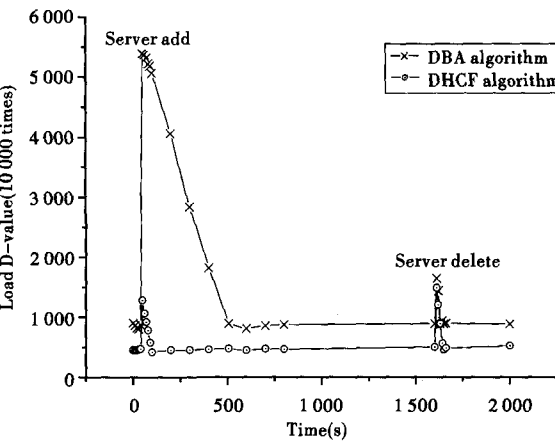


图 4 随机变更服务器时系统稳定性比较

图 3 是在系统压力逐步增加的情况运行两种不同算法得到的结果. 由图 3 可见随着系统压力不断增大, CHDF 算法的负载均衡结果更加稳定.

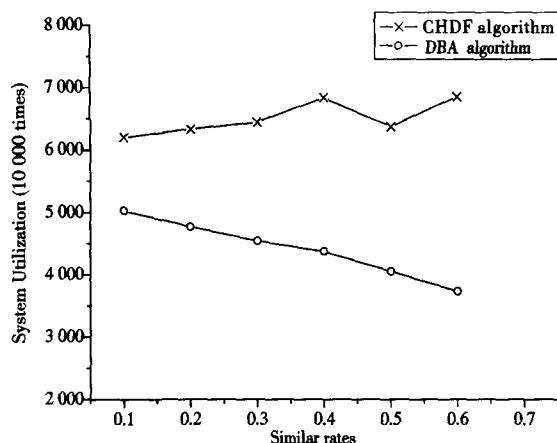


图5 系统负载率与请求相同度比较

图4是系统中随机添加和删除视频服务器的情况下,两种算法的运行结果.由图4可见,CHDF算法相对于通常的动态反馈具有更好的抗冲击性,可以实现视频服务器的热变更.

由于分布式视频监控系统中经常会有同一路视频被多个客户端调看的情况.图5就是在同一路视频调看比例逐步增加的情况下,分别运行两种算法的负载率情况对比图.可以看出,调看相同视频占据的比例越高,CHDF算法的系统整体负载率越低.

5 结束语

本文提出的CHDF算法借鉴了一致性哈希原理和动态反馈机制,实现了一种新的视频请求分配算法,提高了大型视频监控系统的稳定性和抗冲击能力,并根据视频监控系统的业务特点作了优化和改进,节约了系统资源.在越来越普及的大型视频监控系统中具有良好的应用前景.

参考文献:

[1] Karger D. Simple efficient load balancing algorithms for peer-to-peer systems [J]. Theory of Computing Systems, 2006, 39(6): 787-804.

- [2] Idris A. Rai and Murat Alanyali, Uniform Weighted Round Robin Scheduling Algorithms for Input Queued Switches [C]// IEEE ICC 2001. Helsinki, Finland; IEEE, 2028-2032.
- [3] Jong-seon Kim, Daniel C. Lee, Weighted Round Robin Packet Scheduler Using Relative Service Share [C]// Military Communications Conference, 2001. MILCOM, 2001. Communications for Network-Centric Operations; Creating the Information Force. IEEE, 2001.
- [4] Niyato D, Srinilta C. Load Balancing Algorithms for Internet Video and Audio Server [C]// Proceedings of the IEEE International Conference on Networks, [S. L.]; IEEE, 2001.
- [5] George Cybenko, Dynamic Load Balancing for Distributed Memory Multiprocessors [J]. Journal of Parallel and Distributed, 1989, 7(02): 279-301.
- [6] Karger D, Lehman E, et al. Consistent hashing and random trees: Distributed caching protocols for relieving hot spots on the World Wide Web [C]// Proceedings of the 29th Annual ACM Symposium on Theory of Computing, El Paso, Texas; ACM, 1997: 654-663.
- [7] Karger D, A Sherman, et al. Web caching with consistent hashing [J]. Computer Networks 31 (11): 1203-1213.
- [8] Barazandeh I, Mortazavis S. Two Hierarchical Dynamic Load Balancing Algorithms in Distributed systems [C]// Proc of the 2nd international Conference on Computer and Electrical Engineering, Dubai; IEEE, 2009: 516-521.

作者简介:

胡丽聪 女, (1987—), 硕士研究生. 研究方向为通信与信息系统、视频处理.

徐雅静 女, (1977—), 博士, 讲师. 研究方向为数据挖掘、视频处理.

徐惠民 男, (1941—), 教授, 博士生导师. 研究方向为 Internet 及其应用.