P2P Stream Data Allocation Algorithm

Ruifeng Shao

Department of Computer Science and Technology, Dalian Neusoft Institute of Information, Dalian, China

Email: *ruifshao@126.com*

Abstract: At present, streaming media service is one of the important application of the information technology. P2P network has played an important role in the distribution of large data flow of streaming media in application. In this paper, we propose a P2P streaming media data allocation algorithm, which is based on the importance degree of data block and buffer delay. It improves the traditional allocation algorithms from the aspect of buffer delay and streaming media data's continuity, using the greed strategy of allocating the data block in priority according to their importance degree. The experimental results show that compared with other similar allocation algorithms, the improved algorithm's buffer delay is lower and the media data can be continued played better.

Keywords: P2P; streaming media; aata allocation;

P2P 流媒体数据分配算法

邵锐锋

大连东软信息学院，大连，中国，116023

Email: ruifshao@126.com

摘　要：P2P 流媒体数据分配技术是 P2P 流媒体系统的核心技术。本文介绍了一种基于块重要度与缓冲延迟的 P2P 流媒体数据分配算法，采用系统缓存延迟优先分配及数据块重要度优先分配的贪心策略，从缓存延迟、流媒体数据连续性等方面对传统数据分配算法进行了改进。通过仿真实验表明，改进的算法比同类算法具有更小的缓冲延迟，并且保证了媒体数据的连续播放。

关键词：P2P；流媒体；数据分配;

1 引言

P2P 流媒体技术是目前流媒体领域的前沿技术。由于流媒体数据对带宽的高要求，使得流媒体分发在经历了 C/S、IP 组播、内容分发网（CDN）之后，P2P 技术与应用层组播成了主流的技术[1,2]。P2P 流媒体系统在缓冲延迟、数据分配、流数据连续性等方面仍有着许多问题有待解决。主要表现为：一、数据缓冲时间过长，浪费等待播放的时间；二、数据块分配不合理，数据不能连续播放，造成‘卡’的现象；三、缓存的数据块太多，浪费大量的内存。

数据分配，也就是流媒体数据的传输调度策略。在 P2P 流媒体系统中占有十分重要的位置，传统的数据分配基本上属于“稀有数据优先”的原则，主要应用在文件系统中。但是由于流媒体数据的实时性和连续性使得数据分配算法有了更大的研究空间。目前数据分配算法主要分为基于数据包的细粒度调度和基于数据层的粗粒度调度[2]。文献[3]提出了一种结合网络拓扑发现的 P2P 流媒体服务体系 PROMISE,它主要考虑了多对单传输模式下可能出现的共享瓶颈网络带宽对接收方的影响；文献[4]提出了一种自适应的分层 P2P 流媒体框架,它结合网络的状态和接收方分层缓冲区的状态,引入滑动窗口机制,在各个发送方之间进行数据分配,以平滑接收方的服务质量。另外近年来研究人员抽象出了一些分配模型，比如 OTS[5]、FSS[6]、ZBS[7]等。

本文的算法属于细粒度的数据分配算法，以数据块重要度优先及数据的缓冲延迟最小化为目的进行 P2P 流媒体数据分配。主要从缓冲延迟、数据连续性两个方面对比本算法与其他分配算法的性能。

2 系统模型

目标系统通过请求节点 R 构建拓扑，通过对媒体数据和可供应节点带宽的计算，利用数据分配算法得到服务供应节点集合 $P=\{P_1,P_2,P_3,\cdots\cdots,P_n\}$。供应节点可随时加入或离开系统，并且供应节点带宽为异构的。假定媒体数据采用累计分层的编码方式，每层数据分为大小相同的数据块，每隔一定的时间，请求节点会

根据带宽情况请求若干数据块，系统的局部模型图见图 1。

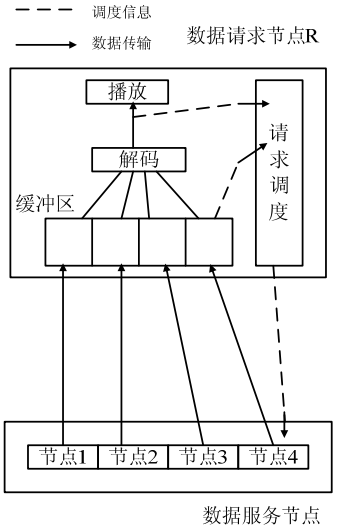

图 1 局部系统模型图

数据分配的协调由数据请求节点实施，大致流程如下：根据数据服务节点带宽和可用数据信息，请求节点 R（设其下行带宽为 R_{in}）周期性的确定希望下载的缓冲内容。然后选择服务节点集合 $P=\{P_1,P_2,P_3,\cdots\cdots,P_n\}$并利用数据分配算法计算分配结果，得到分配给节点 P_i 的数据块集合 $B_i=\{b_k|k\in\{1,2,\cdots,m\}\}$，m 是希望下载的数据块总数。以此形成若干请求列表发给供应节点，供应节点则根据请求列表来控制内容的传输。在接受数据节点一端，缓冲区负责将传输来的数据解码并交给客户端播放，同时利用滑动窗口机制配置缓存区，以实现周期性的质量适配。在周期时间内，根据缓冲区和播放时间等调度信息，请求节点会再次发送请求资源信息，进行下一次数据分配调度。

在 P2P 流媒体数据分配模型中，缓冲延迟 TD 是一个重要的指标。

定义 1 缓冲延迟是指从数据开始传输到数据开始回放的时间间隔。

在本文中，假设 $P_{out}(i)$为服务供应节点 P_i 的上行带宽（输出带宽）。数据 L 分为 m 块(0<m<n)，假设每个数据块的大小相同为 B 且播放时间相同为 R_0，所有的时间都以播放一个数据块所用的时间 δt 为单位，即 $\delta t = B/R_0$ [8]。$P_i(b_k)(1\leqslant k\leqslant m)$表示节点 P_i 负责传输其中一个数据块 b_k，且该节点 P_i 已经负责传输的排序在 b_k 之前数据块个数为 $S_k(P_i)$，则数据块 b_k 实际到达的时间为：

$$t_k=\frac{(S_k(P_i)+1)\times R_0}{P_{out}(i)}，单位是（\delta t） \quad (1)$$

若保证连续播放，缓存延迟 TD 需满足 TD+k-1≥t_k（假设每个数据块全部下载完成后进行播放），即 TD≥t_k-k+1。故缓冲延迟可以表示为：

$$TD=\max\{(t_k-k+1)\,|\,k\in\{1,2,\ldots,m\}\},单位为\ \delta t$$

（2）

在细粒度的 P2P 流媒体分配算法中，数据块是按照编号顺序进行播放的。由此而知，编号越小的数据块对于下载速度的要求越高，即对提供节点的带宽要求越高，这个数据块也越重要。设数据总块数为 m，块重要度用 IOB_i (1≤i≤m)表示，则有如下定义：

定义 2 数据块重要度是指根据该数据块的帧特性和播放顺序而决定它对整个系统的影响程度

$$IDB_i=\frac{\lambda\times(m-i+1)}{m},\lambda是帧特性影响系数$$

（3）

由定义可知，数据块 1 的重要度最大，数据库 m 的重要度最小。

系统分配算法的目标是充分利用服务供应节点的带宽，减少播放缓冲时间，并且保证播放的连续性，使得在动态网络情况下获得高的播放质量。该算法需要考虑一下几个约束条件：

$$R_{in}\leq\sum_{i=1}^{n}P_{out}(i)\wedge\forall i\in\{1,2,\ldots,n\},R_{in}\geq P_{out}(i)$$

（4）

$$\min(TD)\wedge\forall P_{out}(b_i)>P_{out}(b_j),IDB_i>IDB_j$$

（5）

式（4）表示供应节点上行带宽和不小于请求节点下行带宽并且请求节点的下行带宽不小于任何一个供应节点上行带宽。式（5）表示目标算法追求最小的缓冲延迟并且保证优先分配重要度高的数据块，以保证播放质量。

3 算法描述

基于上述数据分配模型，本文算法需要从以下几个方面入手：在提供的供应节点集合中选取带宽最优化的节点，为了降低系统的并行处理难度，节点数目控制在一定范围内（用 n 表示节点数目）；在请求数据节点方面，把请求的数据分成块，并根据式（3）将数据块按照重要度降序排序；按照式（1）计算每个数据块分配给每个供应节点的下载用时，选取合理的服务节点，寻找缓冲时延和重要度的最优组合，实现低时延、连续流畅的高质量流媒体服务。根据以上原则，本文算法主要流程如下：

1）在固定的周期 Δ 内，请求节点 R 确定需要请求的数据块集合 $B=\{b_k|k\in\{1,2,\ldots,m\}\}$，m 是希望下载的数据块总数。

2)按照式(3)计算每个数据块 b_k 的重要度 IOB_k，按照重要度进行降序排序得到排序后的集合 $B_{sort}=\{b_k|IOB_k\leqslant IOB_{k+1}，k\in\{1,2,\ldots,m\}\}$。

978-1-935068-18-1 © 2010 SciRes.

3）在供应节点集合中按照带宽大小选取 n 个节点作为服务提供节点集 P={P_1,P_2,P_3,……,P_n}，按照式（1）计算每个数据块分配给每个服务节点的下载用时，取用时最小的节点作为该数据块的下载节点。

4）重复 3）操作，知道所有数据块都找到合适的下载节点。得到数据分配集 W={ $P_i(b_k)$|i∈{1,2,…,n}，k∈{1,2,…,m}}。

算法的代码描述如下：

Algorithm
Begin
While time≤Δ do
1. PeerR Ask for B:={b_k|k∈{1,2,…,m}}，m 是希望下载的数据块总数；
2. PeerP Choose P:={P_1,P_2,P_3,……,P_n} and $P_{out}(1)$ ≥$P_{out}(2)$≥$P_{out}(3)$≥……≥$P_{out}(n)$;
3. B_{sort}={b_k| IOB_k ≤ IOB_{k+1}， k∈{1,2,…,m}}= Sort(B={b_k|k∈{1,2,…,m}}) by IOB Desc；
4. For k:=1 to m do
(1) i:=1;
(2) For j:=1 to n-1 do
if $t_k(j)$＞$t_k(j+1)$ and 对 P_j 的分配没有完成 then
i:=j;
(3) update W($P_i(b_k)$∈W);//更新分配集
(4) $S(P_i)$:= $S(P_i)$+1;
//更新节点分配数据块数目
//用于计算 $S_k(P_i)$
5. End while
End.

OTS算法

P1	1	2	5	9
P2	4		8	
P3	3		7	
P4	6			

本文算法

P1	1	2	8	9
P2	3		5	
P3	4		6	
P4	7			

图 2 OTS 算法与本文算法数据分配结果

本文算法主要是考虑缓冲延迟最小和保证播放连续性，可以用图表形象的对比该算法与其他分配算法的差别（对比算法为 OTS 分配算法）。假设四个供应节点（P_1,P_2,P_3, P_4），输出带宽依次为 R_{in}/2、R_{in}/4、R_{in}/4、R_{in}/8（满足 OTS 分配算法带宽要求，R_{in} 为请求节点带宽），共有 9 个数据块需要分配，则两个算法的分配结果如图 2 所示。

图中两个分配算法根据式（2）计算缓冲延迟结果分别为 3δt、4δt ，可见 OTS 算法在理论上已经基本上实现了缓冲延迟的最小 。但是 OTS 只是从缓冲延迟方面考虑了算法的性能，而本文的算法则是考虑了重要数据块优先传送，保证了流媒体的播放质量。

4 实验结果与分析

本文实验利用 NS2 软件在 PC 机上建立网络拓扑，从缓冲延迟和播放流畅性两个方面测试 BIC_{P2P} 算法的性能。

实验建立一个拥有 5 个节点的网络拓扑，其中请求节点的输入带宽为 1.5Mbps，另外 4 个供应节点的输出带宽分别为 700kbps、400kbps、400kbps、200kbps。请求数据为 700M，分为 100 个数据块，则每个数据块的大小为 7M，流媒体采用 MPEG-1 编码，假设提供节点没有失效的前提下，OTS 算法和 BIC_{P2P} 算法的缓冲延迟及播放质量的对比情况见图 3 和图 4，其中图 4 利用在固定的 60 秒时间内的播放等待次数来反应算法的播放质量。

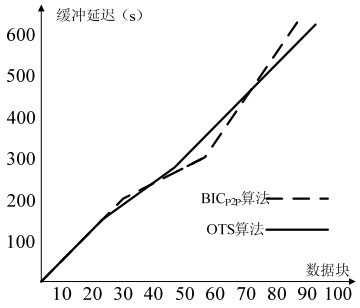

图 3 不同数据块缓冲延迟对比

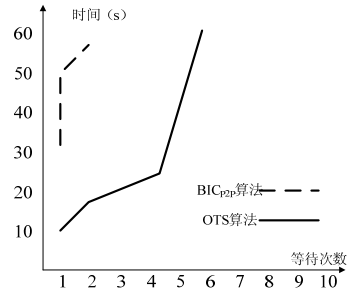

图 4 固定时间内播放等待次数

如图所示，BIC_{P2P} 算法在滴缓冲延迟的情况下，保证了流媒体的播放质量，主要由于该算法利用块重要度优先的策略，使得对于播放重要的数据块能够选择带宽资源丰富的节点进行数据传输。对比之下，

BIC_{P2P} 算法比 OTS 算法在播放质量方面有更好的保证，会极大提高系统的性能。

五 结束语

本文利用缓冲延迟以及数据块重要度设计了一个保证小缓冲延迟和播放质量的 P2P 分配算法。在细粒度的调度算法中，多数算法只强调数据的缓冲延迟，而忽略了流媒体的播放质量，实际应用价值不高。而且很多算法对输出带宽有定性要求，极大的限制了系统的动态性。仿真实验表明，本文提出的算法能够使系统各个节点获得优化的服务启动时间，同时保证流媒体分发的服务质量。

References (参考文献)

[1] Yu Sheng sheng, Hu Wen bin. P2P scheme for he media stream multicast based on density tree[J]. Micro Systems, 2008. Vol. 27(11), pp: 2020-2024.

[2] Dd Jin B Kwon, Heon Y Yeom. Distributed multimedia streaming over peer to peer networks[M]. Parallel Processing, Austria: Springer BerlinHeidelberg, 2007.

[3] Hefeeda M, Habib A, Botev B, Xu D, Bhargava DB. PROMISE: A peer-to-peer media streaming system. In: Lawrence AR, ed. Proc. of the ACM Multimedia 2003. New York: ACM Press, 2003, pp: 345-351.

[4] Rejaie R, Ortega A. PALS: Peer-to-Peer adaptive layered streaming. In: Christos P, Kevin CA, eds. Proc. of the ACM NOSSDAV 2003. New York: ACM Press, 2006.

[5] Xu Dongyan, Mohamed Hefeeda. On peer to peer media streaming[C]. Proceedings of the 22nd Iternational Conference on Distributed Com puting Systems (ICDCS'08), USA: IEEE Computer Society, 2008, pp:363-368.

[6] Jin B Kwon，Heon Y Yeom. Distributed multimedia streaming over peer to peer networks[M]. Euro—Par 2003 Parallel Processing, Austria: Springer Berlin, Heidelberg, 2004, pp:851-857.

[7] Dong Hai tao. Theoretical analysis and system study on peer to peer media streaming[D]. Tsinghua University, 2005.

978-1-935068-18-1 © 2010 SciRes.