ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.17, No.4, April 2006, pp.915–924
DOI: 10.1360/jos170915

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

支持高并发度的大容量媒体库*

陈 明+, 杨广文, 王鼎兴

(清华大学 计算机科学与技术系,北京 100084)

Large-Capacity Media Library Supporting Highly Simultaneous Access

CHEN Ming+, YANG Guang-Wen, WANG Ding-Xing

(Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China)

+ Corresponding author: Phn: +86-10-62799645, E-mail: cm01@mails.tsinghua.edu.cn

Chen M, Yang GW, Wang DX. Large-Capacity media library supporting highly simultaneous access. *Journal of Software*, 2006,17(4):915–924. http://www.jos.org.cn/1000-9825/17/915.htm

Abstract: Large-Capacity media libraries supporting highly simultaneous access are more and more popular. Media libraries based on traditional client/server mode, or pure Grid or P2P can hardly meet the requirements of both high concurrency and reliable service. This paper proposes a new architecture combining Grid and P2P for media library—NeoMedia. It consists of the dedicated and volunteered nodes supported by the system server and forms a virtual media pool of large capacity. Coordinated by the system server, nodes requesting files from the media pool serve each other in the style of P2P, which further enhances the system's performance. According to access pattern and intensity, the system server automatically adjusts the allocation of resources, adaptively optimizing the system's overall performance. NeoMedia targets at view-after-downloading, which can efficiently utilize servers' precious bandwidth and clients' power. Theoretical analysis demonstrates that NeoMedia is able to support hugely simultaneous requests and provide non-trivial services at the same time with relatively low server-side bandwidth consumption.

Key words: P2P; grid; media library; highly simultaneous access

摘 要: 支持大容量和高并发度的媒体库越来越流行.利用传统的基于纯粹的服务器/客户端、对等网络或网格的方法构造这种类型的媒体库,难以同时满足高并发度和可靠服务的要求.提出了一种结合了网格和对等网络(peer-to-peer)的媒体库架构——NeoMedia.专用的和志愿参与服务的节点在系统服务器的支持下,形成了一个大容量的虚拟媒体存储池,请求服务的节点在从媒体存储池下载媒体文件的同时,在系统服务器的协调下,采用对等网络的方式相互提供服务,从而进一步提升系统的性能.系统服务器还根据虚拟媒体池的负载模式和强度的变化自动调整服务的资源分配,自适应地优化系统的性能.NeoMedia 面向的是下载完毕再观看这种可以充分利用服务器的带宽和客户机能力的模式.理论分析表明:系统能够在使用较少的系统带宽的情况下,支持巨大的并发的用户请求,并且提供非平凡服务.

* Supported by the National Natural Science Foundation of China under Grant Nos.60373004, 60373005, 90412006, 90412011, 60573110 (国家自然科学基金); the National Grand Fundamental Research 973 of China under Grant No.2004CB318000, 2003CB316907 (国家重点基础研究发展规划(973))

Received 2005-04-04; Accepted 2005-08-03

关键词: 对等网络;网格;媒体库;高并发度
中图法分类号: TP393 文献标识码: A

随着互联网的发展,支持大容量和高并发度的媒体库是当前的一个研究热点.这些媒体库通常要求支持 Tbytes 级的存储以及 1000 以上的在线用户.在这样的负载下,即使每个在线用户消耗 512Kbps(一个很小的、只能播放低质量视频和音频的码率)的服务器带宽,系统的总带宽消耗也超过了 512Mbps.这对目前的单个服务器能力和网络带宽都构成了严重的挑战.

有的方案采用集群技术(DALA)[1]来提高服务器的处理能力,或代理服务器技术(MiddelMan)[2]来减少服务器带宽的消耗.这些方案都面临着服务器/代理服务器出口带宽瓶颈的问题.有些采用基于 IP 的组播技术方案(patching)[3−5]虽然解决了服务器出口带宽的问题,但是要求网络必须支持 IP 组播.这个要求对现有的网络是不现实的.

伴随计算机科学和技术的发展,个人计算机的性能依循莫尔定律(每隔 18 个月翻一番)而快速发展着.存储和网络带宽的发展速度超越了莫尔定律.现代个人计算机的能力已经大大超过了 20 年前的超级计算机.这两个趋势的发展使得构造基于 P2P 的应用层的组播(OStream[6],SplitStream[7])和内容分发网络(CDN) (Bullet[8], Slurpie[9],Bit Torrent[10])成为可能.这些研究主要集中在数据的高效分发上,没有涉及到数据的管理.

P2P 的文件共享系统可以看作媒体库的一个特例,例如 KaZaa[11]和 Gnutella[12]等.在这些系统中,每个参与者都共享自己的一部分文件和开放自己的一部分上载带宽,以向其他参与者提供下载服务.这是一个松散无中心的结构,没有数据的管理,查找所需的媒体资源较为困难.系统不保证服务质量:参与者一般都是普通的个人电脑,从单个来看能力较弱,同时对自己所提供服务的质量也不作任何保证.研究[13]表明,这些系统里面大部分都是热门文件,冷僻的很少;查找和下载非热门媒体文件较为困难.

本文提出了结合了网格和对等网络的媒体库架构——NeoMedia:专用的和志愿参与服务的节点在系统服务器的支持下,形成了一个大容量的虚拟媒体存储池.请求服务的节点在从媒体存储池下载媒体文件的同时,在系统服务器的协调下,采用对等网络的方式相互提供服务.系统服务器还根据虚拟媒体池的负载模式和强度的变化自动调整服务的资源分配,自适应地优化系统的性能.NeoMedia 引入在中心服务器管理下的虚拟媒体库和协调下的基于对等网络的数据散发模式,给系统带来了传统服务器/客户端模式下的可靠性和可用性,以及对等网络的可扩展性和经济性.系统除了提供大容量的媒体库以外,还提供非平凡的支持高并发度的下载服务.

NeoMedia 面向的是下载完毕再观看这种模式.尽管这种模式相对于边下载边观看模式有一定的延迟,但却可以有效地利用服务器的宝贵带宽和客户机的能力,并且可以保证观看的连续性等播放的质量.

1 系统架构

本节首先简要介绍 NeoMedia 用到的垂直下载,然后给出 NeoMedia 的详细设计,包括系统结构和关键的自适应调整算法.

1.1 垂直下载(**perpendicular downloading**)

垂直下载是对等网络使用的一种新技术,它可以极大地提高系统的吞吐量和可扩展性.很多系统,例如 Bullet[8],Surpie[10],Bit Torrent[11],采用垂直下载来提高系统的性能.其典型的工作模式如下:

假定 A 为某个节点,F 为拥有某个完整的文件.一群下载节点要从 A 节点下载 F.A 把 F 分解成许多固定大小、连续的块(称为下载分块).这些块相互不相交,它们的并集等于 F.下载节点从 A 上随机地选择自己还没有的块下载.那么任意两个下载节点之间,它们已经下载完毕的块在很大程度上可能是互补的.也就是说,它们相互拥有对方需要的块.此时,一个下载节点除了从 A 上继续下载需要的块以外,还从其他下载节点已经下载了的、并且自己需要的块当中随机地选择块下载.下载节点继续这个过程,直到自己把 F 的分块完全下载完毕,可以重建出 F 为止.由此可见,系统中存在多条下载通道:除了从 A 到下载节点群的下载通道以外,还有许多下载节点之

间的下载通道,可以把前者看成与后者是垂直的.当有多个节点共享同一个文件 F 的时候,工作模式和只有一个节点共享 F 的模式也是类似的:每个共享 F 的节点都对 F 依据系统指定的块大小进行分块,因此,每个共享节点分解出来的块是相同的.下载节点可以向一个或多个共享节点请求块.

在这种模式下,除了 A 的带宽以外,下载节点之间的带宽也被充分利用起来,极大地提高了系统的性能.垂直下载在构造链接拓扑的时候一般采用“网(mesh)”而不是“树(tree)”的方式,使得构造高效的拓扑变得容易和可行.理论分析和实际测量表明[14]:当系统中有充裕的节点共享文件时,在稳态下,系统中每个下载节点的下载速度能达到 $u--u$ 是每个节点的上传带宽(假定节点的下载带宽大于节点的上传带宽,并且节点停留的时间较长),并且和系统的规模,即下载节点的个数无关.NeoMedia 采用的垂直下载方式是 Bullet[8].

1.2 架构和原理

NeoMedia 提供了一个巨大的媒体资源库供用户下载媒体文件.库中的文件基本上是 100MB 级别的,标准大小是 740MB,即一张光盘的容量.NeoMedia 面向的是拥有宽带连接的用户.NeoMedia 能够在使用较少的系统带宽的情况下支持巨大的并发的用户请求,并且提供非平凡的下载服务.

系统由以下 3 部分组成(如图 1 所示):系统管理与协调服务器(system managing and coordinating server,简称 SMCS)、存储节点群、下载节点群.存储节点在系统服务器的管理下形成了一个大容量的虚拟媒体存储池,下载节点在从媒体存储池下载媒体文件的同时,在系统服务器的协调下,采用改进的垂直下载方式相互提供下载服务.

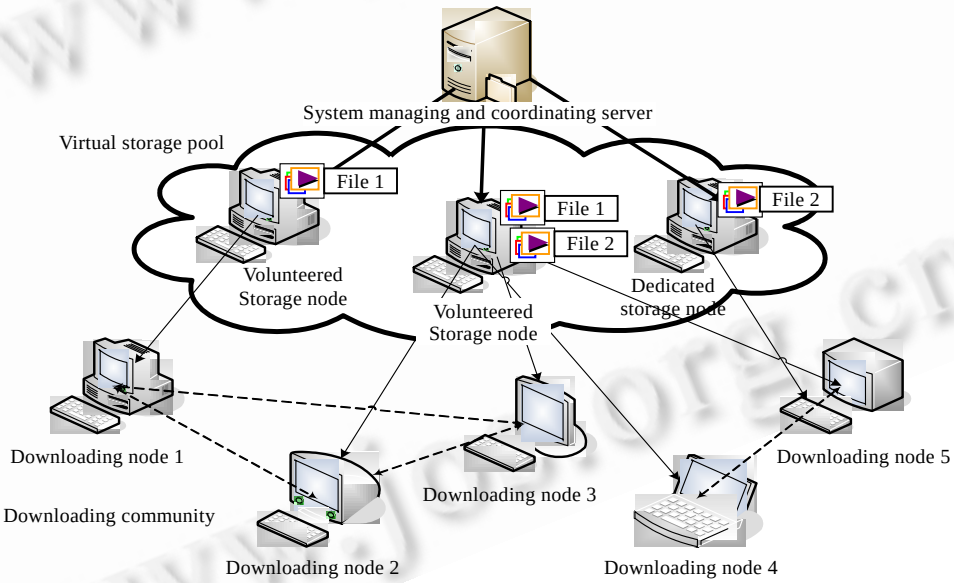


Fig.1 System architecture

图 1 系统结构

存储节点由系统专用节点和志愿参与节点构成.存储节点要求拥有较大的存储空间、较高的联网带宽以及较稳定的在线服务时间.每个存储节点都向系统贡献出一部分硬盘空间.贡献出的硬盘空间在 SMCS 的管理下形成一个比较可靠的巨大虚拟存储池,保存媒体库的媒体文件.虚拟存储池直接提供的带宽,即所有存储节点贡献的带宽之和,称为系统带宽.

媒体库中的每个文件都拆分成固定大小的分块,称为文件分块.文件分块是下载分块的整数倍大小.保存有一个文件的文件分块中第 1 块的存储节点,称为这个文件的宿主节点.每个存储节点维护本地保存的并且完好的文件分块的列表、记录文件的访问频度以及自己的负载.每个文件的宿主节点还为这个文件维护一个实时更新的、正在下载这个文件的下载节点的列表.采用文件分块的好处是,在文件访问不均匀的情况下,容易做到存

储节点的负载平衡,提高对热点文件的服务速率.

志愿参与的存储节点由于各种原因可能暂时或永久离开系统.存储节点定期地向 SMCS 发送心跳(heartbeat)包,汇报自己的存在.存储节点每过一定的心跳汇报周期,还必须向 SMCS 汇报自己存储完好的文件分块列表.据此,SMCS 可以维护一个全局存活的存储节点列表、一个文件分块的副本数目列表以及一个从媒体文件到所宿主节点的映射表.SMCS 根据副本数目列表,自动维护系统中每个文件分块的副本数目,保证每个文件都有一定的可用性.虚拟存储池是一个几乎只读的存储系统,对一致性的要求较低.副本的管理主要集中在可用性和性能上.本文主要研究服务的自适应调整问题,我们把副本的可用性维护工作留到未来进行.同时,SMCS 还根据存储节点周期性汇报的每个文件的访问频度和系统的负载,自动调整副本的数目,提高系统的性能.SMCS 周期性地向宿主节点通报保存了这个文件的文件分块的存储节点的列表.

系统典型的工作模式是:下载节点在下载文件之前,先向 SMCS 查询自己感兴趣的文件的宿主节点,SMCS 根据映射表和存活存储节点列表返回符合要求的宿主节点;下载节点据此向返回的宿主节点请求正在下载自己感兴趣文件的下载节点列表和存储了这个文件的存储节点列表;得到这个列表后,下载节点采用垂直下载的方式下载文件.为了减轻存储节点的负载,NeoMedia 采用改进的垂直下载方式:当某个分块被一个或多个下载节点拥有时,下载节点将向拥有此分块的其他下载节点请求此分块,而不向存储节点请求此分块.存储节点也拒绝对此分块的请求,直到此分块再次只有存储节点拥有时为止.所有下载节点的下载速率之和称为系统的总吞吐率.

人们可能会认为 SMCS 是系统的瓶颈,或是系统的单一失效点.事实上,SMCS 的引进极大地简化了系统的设计,并且简单、可靠地解决了纯粹的对等网络中,由于无中心而带来的一系列难以解决的问题,例如全局知识等.SMCS 只做元数据的管理和维护,并不负责数据的传输.GFS[15]表明,这样的一个元数据服务器可以支撑非常大的数据流量.为了进一步提高 SMCS 的可靠性,可以采用机群或多点服务等成熟的可靠性解决方案.SMCS 在很大程度上是一个只读的、并且对数据的实时性和一致性要求不高的组件,它的元数据完全可以从存储节点中恢复过来.SMCS 没有任何设计妨碍采用任何可靠性解决方案.

下面,我们将讨论如何通过调整文件的副本数目来提高由存储节点通过聚合提供的服务速率.

1.3 服务自适应调整

根据存储节点汇报的信息,SMCS 知道存储节点贡献的带宽、文件的访问频度和长度以及系统的负载等信息.在文件分块存储,并且分块在存储节点中分布比较均匀的情况下,可以认为虚拟存储池对一个文件能够直接输出的最大系统带宽(即仅仅从存储池下载文件的速度)和保存这个文件的存储节点个数(也就是副本数)成正比.那么,系统需要计算一个最优化的副本配置方案,使得系统在保证公平性的同时,性能能够最高.即所有下载节点(无论是下载热门文件还是冷僻文件的节点)的下载速度相同,并且最大.

1.3.1 单个文件消耗的系统带宽

我们首先考虑库中只有一个文件的情况.

假定节点 A 共享了一个文件 F.下载节点采用改进的垂直下载方式来下载 F.当 F 的某个分块在下载群中缺失时,A 必须响应对这个分块的请求.现在来求 A 的带宽消耗.推导符号如下定义:

L:文件长度;l:每个下载节点平均下载的长度,$l \le L$.下载完 l 后,这个下载节点将离开系统;d:下载节点平均下载速率.一般节点的下载带宽都大于或等于上载带宽(ADSL 的下行带宽一般大于上行带宽;LAN 则两者相等),根据文献[14]可知,d 近似等于下载节点的平均上传带宽;λ:对这个文件请求的到来速率,单位是个/秒,也是请求这个文件节点的到来速率.可以认为到来速率服从泊松分布;B_c:存储节点消耗的带宽,$B_c=f(L,l,d,\lambda)$.

OStream[6]在下载节点在下载时是顺序下载的假定下——即下载节点在长度为 L 的文件中随机选择连续的 l bytes 下载,得出存储节点消耗的带宽.我们用类似 OStream 的模型推导在改进的垂直下载模型下,存储节点消耗的带宽:

令 x 是 L 中的某个 byte 节点 P_1 在 t 时刻由于下载节点群中没有 x 而向存储节点请求了 x.由于 P_1 对分块的请求是随机的,P_1 请求 x 这个事件在 P_1 在系统停留的时间内是均匀分布的,则 x 由于 P_1 而停留在下载节点群

的时间期望为 $T=l/2d$.在$[t,t+T]$这段时间内,P_1 可以满足其他下载节点对 x 的请求.这个“链条”可以继续下去,直到某次对 x 的请求由于前一个请求过 x 的节点离开了系统中断为止.令 X 表示一个对 x 的请求事件,ω表示相邻的两个 X 的时间间隔,Z 表示存储节点收到的对 x 的请求,显然,$\{Z\}\subseteq\{X\}$,τ为相邻的两个 Z 的时间间隔.

节点的到来服从泊松分布,对分块的请求是随机、均匀地分布的,则$\{X\}$也是一个泊松流,它的到达速率为 $\lambda_x=\lambda l/L$.

$\omega\leq T$ 的概率为 $F_\omega(t)=P\{\omega\leq T\}=1-\mathrm{e}^{-t\lambda_x}$.

ω的概率密度函数为 $f_\omega(t|x)=\dfrac{\mathrm{d}F_\omega(t)}{\mathrm{d}t}=\lambda_x\mathrm{e}^{-\lambda_x t}$.

现在计算当$\omega\leq T$ 时,ω的期望: $E_{\omega\leq T}(\omega\mid x)=\dfrac{\int_0^T tf_\omega(t\mid x)\mathrm{d}t}{P\{\omega\leq T\}}$.

同样可以计算,当$\omega>T$ 时,ω的期望: $E_{\omega>T}(\omega\mid x)=\dfrac{\int_T^\infty tf_\omega(t\mid x)\mathrm{d}t}{P\{\omega>T\}}$.

令 $p=P\{\omega\leq T\}$,那么 X 可以连续$(n-1)$次从下载节点中取得,而最后一次只能从存储节点中取得的概率是 $p_n=p^{n-1}(1-p)$;这连续 n 次的时间长度的期望为 $t_n=(n-1)E_{\omega\leq T}(\omega|x)+E_{\omega>T}(\omega|x)$.这是两个相邻的、间隔了$(n-1)$个 X 事件的 Z 事件时间长度的期望.

那么,τ的期望为 $E(\tau\mid x)=\sum_{n=0}^{\infty}t_n\times p_n$.

τ表示存储节点收到的、对 x 的请求的时间间隔.所以,存储节点消耗的带宽为

$$B_c=\int_0^L\frac{1}{E(\tau\mid x)}\mathrm{d}x=\frac{\lambda l\left(\mathrm{e}^{\frac{\lambda l^2}{2Ld}}-1\right)}{\mathrm{e}^{\frac{\lambda l^2}{Ld}}-\frac{\lambda l^2}{2Ld}-1} \tag{1}$$

1.3.2 所有文件消耗的系统带宽

本节把上一节的分析扩展到多个文件的情况,求解库中有多个文件时消耗的系统带宽.

文献[16]研究表明,对媒体文件和对网页的请求频度都服从 ZipF 分布.也就是说,把文件按照访问频度由高到低排列,依次编号为 1,2,3,…,那么,第 i 号文件的相对访问频度为 $P_i=\dfrac{1}{z^i\sum_{j=1,2,\ldots}z^{-j}}$,其中 z 为 ZipF 分布的系数.在本系统中,同样可以认为下载节点对库中文件的相对访问频度服从 ZipF 分布.重新定义推导符号:

F:库中文件的数目;

L:库中文件的平均大小;

l:下载节点下载一个文件时平均的下载长度;

λ:对库文件的请求的到来速率;

B_c:所有文件消耗的系统带宽;

U:系统在线用户数(正在下载的用户数);

$B_{throughput}$:系统的总吞吐率.

第 i 个文件的绝对访问频度为

$$\lambda_i=\frac{\lambda}{z^i\sum_{j=1,2,\ldots}z^{-j}} \tag{2}$$

根据式(1),它消耗的带宽为

$$B_i=f(L,l,d,\lambda_i) \tag{3}$$

根据式(1),总共消耗的带宽为

$$B_c(F,l,d,\lambda)=\sum_{i=1}^{F}\frac{\lambda_i l\left(e^{\frac{\lambda_i l^2}{2Ld}}-1\right)}{e^{\frac{\lambda_i l^2}{Ld}}-\frac{\lambda_i l^2}{2Ld}-1} \tag{4}$$

1.3.3 模型求解

现在,我们来求解系统所能支撑的最大下载速度 $d_{\max}$ 和与此对应的每个文件的相对副本数.

依据式(4),如果已知系统带宽 B 和请求的到来速率 λ,则可以求出此时系统所能支持的最大下载速率 $d_{\max}$:

$$d_{\max}(B,L,l,\lambda)=\arg\{B_c(L,l,d,\lambda)=B\} \tag{5}$$

由于式(4)太复杂,我们只求它的数值解.很显然, $B_c(F,l,d,\lambda)=\sum_{i=1}^{F}\frac{\lambda_i l\left(e^{\frac{\lambda_i l^2}{2Ld}}-1\right)}{e^{\frac{\lambda_i l^2}{Ld}}-\frac{\lambda_i l^2}{2Ld}-1}\approx\sum_{i=1}^{F}\frac{\lambda_i l e^{\frac{\lambda_i l^2}{2Ld}}}{e^{\frac{\lambda_i l^2}{Ld}}}=\sum_{i=1}^{F}\frac{\lambda_i l}{e^{\frac{\lambda_i l^2}{2Ld}}}$ 是一个关于 d 的光滑的增函数.采用牛顿法可以快速求出 d 的数值解.

求出 $d_{\max}$ 后,依据式(3),可以得出在最大下载速率下每个文件消耗的系统带宽,从而得出为了支持这个带宽所需要的相对副本数:

$$\frac{r_i}{\sum_{j=1}^{F}r_j}=B_c(L,l,d_{\max}(B,L,l,\lambda),\lambda_i)/B \tag{6}$$

根据 Little 公式,得出在线用户数:

$$U=\lambda\times l/d \tag{7}$$

系统的总吞吐率为

$$B_{throughput}=U\times d=\lambda\times l \tag{8}$$

由式(8)可见,NeoMedia 的总吞吐量随着请求速率的增大而相应地呈线性增长.

2 初步的分析和实验

本节根据上一节得出的系统模型,分析系统在典型应用中的性能.

我们合理地假定系统平均有 100 台存储节点在线,每个存储节点贡献出 20G 的硬盘空间和 100KBytes/s 的下载带宽.在此情况下,虚拟存储池的系统带宽为 B=100×100KB/s=10MB/s,一个典型的 100Mbps 以太网服务器能提供的下载速率,容量为 2TB.媒体库中有 F=1000 部电影,每部电影的平均大小为 L=740Mbytes,客户端下载一部电影时平均下载的长度 l=L×0.95=703Mbytes(某些用户可能中途放弃下载).z 取 0.7[17].如果采用 FTP 方式,系统能够支持的最大到来速率为 $\lambda_{FTP\_\max}=l/B\approx 0.015$.

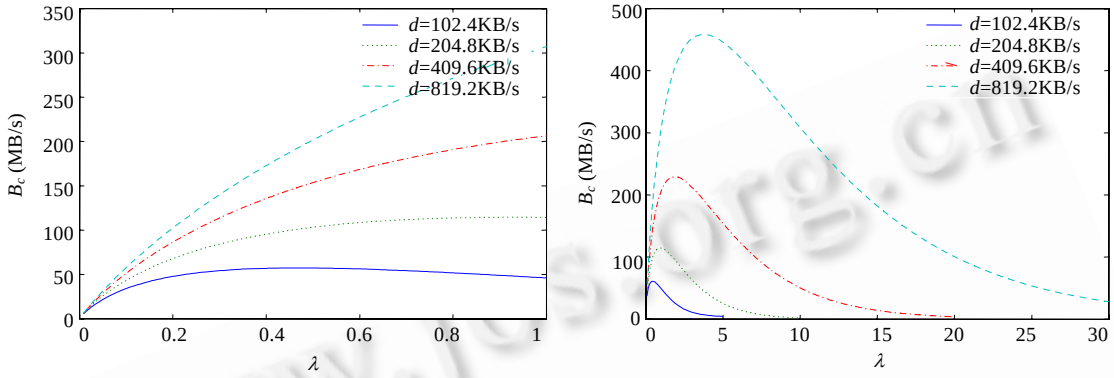
2.1 消耗的带宽

固定 d,根据式(4)求解不同的到来速率 λ 下消耗的系统带宽 B_c(假定 B 无穷大),如图 2 所示.

由此可见,NeoMedia 在不同请求速率下消耗的系统带宽和传统的基于服务器/客户端模型(例如 FTP/HTTP)消耗的带宽完全不一样.在传统的服务器/客户端模型中,如果要对每个请求维持一定的服务速率,那么请求速率越高,消耗的系统带宽也越大.而在 NeoMedia 中,消耗的带宽随着请求速率的增加而到达最大值后,随着请求速率的增加反而减少.这一现象可以这样解释:在下载节点下载速率 d 不变的条件下,依据式(6),随着到来速率的增大,系统中的下载节点随之增多,则一个分块在下载节点中存在的概率随之增大,需要向存储节点请求分块的频度下降,消耗的系统带宽相应减少.

图 3 展示了在不同到来速率下库中不同文件消耗的系统带宽.在传统的服务器/客户端模型中,相对访问频度越高的文件,即 i 越小的文件,它们消耗的带宽越大.NeoMedia 表现了不同的工作特性上:当 λ 较小时,相对访问频度较低的文件几乎没有被访问,这时带宽开销主要集中在相对访问频度较高的文件上;λ 在达到一定大小时,

相对访问频度较高文件的绝对访问速率较大,访问这些文件的下载节点可以相互补充需要的块,它们消耗的系统带宽比较少;相对访问频度中等的文件有一定的绝对访问速率,但又未大到访问这些文件的下载节点可以相互补充需要的块的地步,它们主要向存储节点请求块,因此消耗了相对较多的系统带宽;相对访问频度较低的文件的绝对访问速率仍较小,消耗的带宽也较少;而当λ很大的时候,相对访问频度较低的文件虽然有较大的绝对访问速率,但它们相对较热门的文件来说,其相对访问数依然比较少,消耗的系统带宽也相对较多.



(a) Consumed bandwidth when λ is low (b) Consumed bandwidth when λ is high

(a) λ较小时消耗的带宽 (b) λ较大时消耗的带宽

Fig.2 Consumed bandwidth

图 2 消耗的带宽

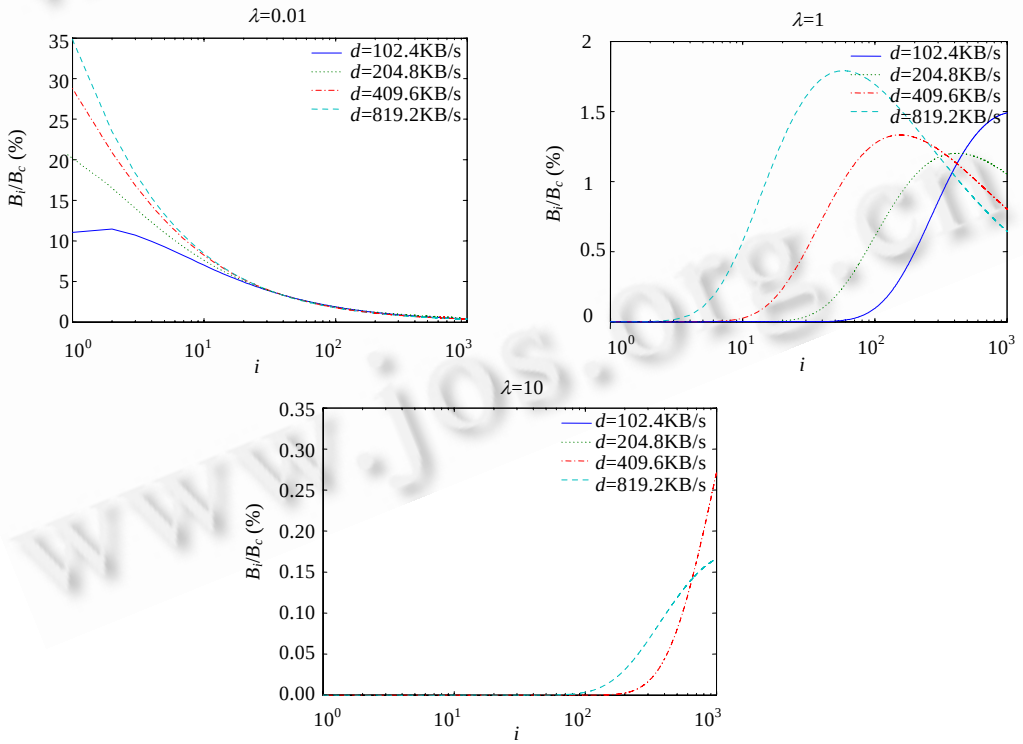


Fig.3 Distribution of consumed bandwidth of files

图 3 库中文件消耗的带宽分布

2.2 下载速率

固定 B,根据式(5)求解到来速率λ不同时,系统所能支撑的最大 $d_{\max}$. ◇ .(I II

NeoMedia 仍然表现出与传统的基于服务器/客户端不同的响应特性:请求的到来速率越大,系统所能提供的最大下载速率越大.在到来速率为 20 个/s 的时候,系统能够支撑超过 400KB/s 的单个用户的下载速率,在线用户超过 20 000;这时系统的总吞吐量超过 8GB/s.这依然可以用改进的垂直下载方式来解释:到来的绝对速率越大,下载节点间可以互补块的概率越大,系统带宽则可以用来进一步提升下载的速率.如图 4~图 6 所示.

不同到来速率下的最佳副本分配也表现了 NeoMedia 的特点:相对访问频度高的文件相对副本数要求较低.

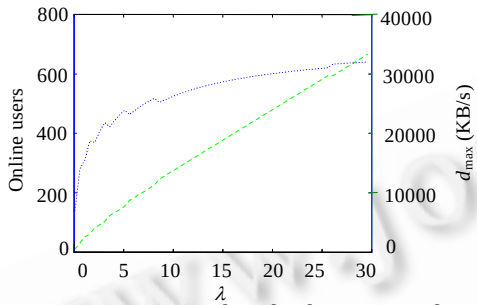


Fig.4 Maximum downloading rates and online users varying arrival rate

图 4 不同的到来速率下系统的最大下载速率和相应的在线用户数

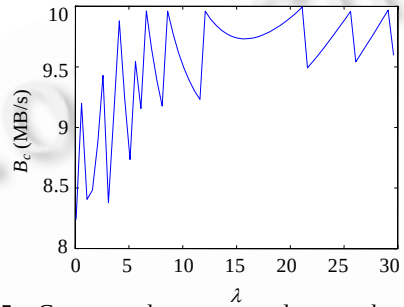


Fig.5 Correspondent consumed system bandwidth (Solved by Newton with step of 50KB/s)

图 5 相应的系统消耗的带宽(采用牛顿法求解,步进精度为 50KB/s)

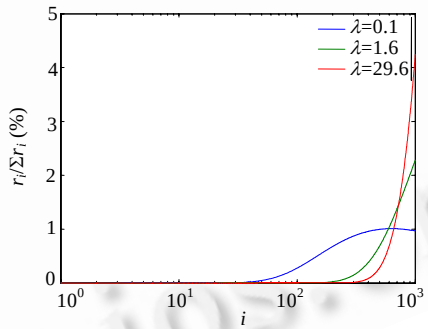


Fig.6 Optimal replicas supporting maximum downloading rate

图 6 支持最大下载速率的最佳的副本分配

3 相关工作

相关工作可以粗分为 3 类:

(1) 传统的基于服务器/客户端的方式,包括 FTP,HTTP,DALA[1]等.服务器拥有较大的存储空间、较大的下载带宽和较强的处理能力.服务器存储和管理所有的媒体资源,向客户提供下载服务.在用户比较少的情况下,可以保证较好的服务质量.同时也可以作为媒体库,保存历史媒体.它们的缺点是:对服务器要求较高、可扩展性弱、应对突发下载的能力较差.DALA 是一个基于集群的结构,通过在集群中的节点自适应分配文件来平衡负载和提高吞吐率.MiddelMan[2]是一个全分布的代理服务器结构,没有考虑数据的自适应调整问题.

(2) 基于 IP 层或应用层组播的方式:OStream[6],Bullet[8],SplitStream[7],Slurpie[9],Bit Torrent[10].这些系统主要研究构造高效的、与网络拓扑相适应的数据散发树(tree)或网(mesh):OStream 的客户端缓存一小段内容,一个客户端只有在其他客户端没有所需的内容时才向服务器请求,以此来减轻服务器的负载;Bullet 在搭建一棵以源

为根的广播树的基础上,再搭建一个网(mesh)来提高广播的质量与效率;SplitStream 则通过搭建多棵子树来分别广播正交的内容——任一节点只在一棵子树中充当非叶子节点,在其他子树中都充当叶子节点;Slurpie 和 Bit Torrent 则通过搭建随机 mesh、通过节点本身的下载速率选择最佳下载节点集的方法来加快下载速度.它们的工作和本文是互补的,可以用来支持 NeoMedia 的基于对等网络的数据散发方法.

(3) P2P 文件共享网络:KaZaa[11]和 Gnutella[12]等.KaZaa 通过系统中自动选出的超级节点(super node)把所有的节点组织起来,提供搜索和下载等服务;Gnutella 中则没有超级节点,采用在 TTL(生存期)限制下的随机漫步和广播的方式来搜索.这些系统中没有专用的服务器,所提供的服务难以保证质量.

4 总结及未来的工作

传统的基于纯粹的服务器/客户端、对等网络或网格构造的媒体库难以同时满足高并发度和可靠服务的要求.本文提出了结合网格和对等网络的媒体库架构——NeoMedia:专用的和志愿参与服务的节点在系统服务器的支持下,形成了一个大容量的虚拟媒体存储池,请求服务的节点在从媒体存储池下载媒体文件的同时,在系统服务器的协调下采用对等网络的方式相互提供服务.系统服务器还根据虚拟媒体池的负载模式和强度的变化,自动调整服务的资源分配,自适应地优化系统的性能.

本文详细分析了 NeoMedia 的性能,得出了系统的性能随着请求速率的增加而基本增强的结论.理论模型还给出了响应服务请求的最佳的文件副本分配方法.

进一步的工作包括:考虑文件可靠性和改变副本分配导致的传输开销下副本的优化方法,以及一个实际运行的系统.

References:

[1] Ge ZH, Ji P, Shenoy PJ. A demand adaptive and locality aware (DALA) streaming media server cluster architecture. In: Proc. of the 13th Int'l Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV 2002). Miami: ACM Press, 2002. 139−146. http://portal.acm.org/citation.cfm?id=507690&coll=Portal&dl=GUIDE&CFID=67945259&CFTOKEN=25416258

[2] Acharya S, Corporation I, Smith B. MiddleMan: A video caching proxy server. In: Proc. of the 10th Int'l Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV 2002). Chapel Hill: ACM Press, 2000. 121−130. http://www.nossdav.org/2000/abstracts/16.html

[3] Hua KA, Cai Y, Sheu S. Patching: A multicast technique for true video-on-demand services. In: Proc. of the ACM Multimedia'98. Bristol: ACM Press, 1998. 191−200. http://portal.acm.org/citation.cfm?id=290771&coll=Portal&dl=ACM&CFID=67945259&CFTOKEN=25416258

[4] Gao LX, Towsley D. Supplying instantaneous video-on-demand services using controlled multicast. In: Proc. of the IEEE Int'l Conf. on Multimedia Computing and Systems. Washington: IEEE Computer Society, 1999, II-2-2, 117−121. http://ieeexplore.ieee.org/search/selected.jsp?qry=%28supplying+instantaneous+video-on-demand+services+using+controlled+multicast%3Cin%3Emetadata%29&srch=1&resset=1331196&imageField.x=69&imageField.y=11&imageField=View+Selected+Items&chklist=778179%40ieeecnfs

[5] Eager D, Vernon M, Zahorjan J. Minimizing bandwidth requirements for on-demand data delivery. IEEE Trans. on Knowledge and Data Engineering, 2001,13(5):742−757.

[6] Cui Y, Li BC, Nahrstedt K. oStream: Asynchronous streaming multicast in application-layer overlay networks. IEEE Journal on Selected Areas in Communications, Special Issue on Recent Advances in Service Overlays, 2004,1(22):91−106.

[7] Castro M, Druschel P, Kermarrec AM, Nandi A, Rowstron A, Singh A. SplitStream: High-Bandwidth multicast in cooperative environments. In: Proc. of the 19th ACM SOSP. Bolton Landing: ACM Press, 2003. 298−313. http://portal.acm.org/citation.cfm?id=945474&coll=Portal&dl=ACM&CFID=67945259&CFTOKEN=25416258

[8] Kostic D, Rodriguez A, Albrecht J, Vahdat A. Bullet: High bandwidth data dissemination using an overlay mesh. In: Proc. of the 19th ACM SOSP. Bolton Landing: ACM Press, 2003. 282−297. http://portal.acm.org/citation.cfm?id=945473&coll=Portal&dl=ACM&CFID=67945259&CFTOKEN=25416258

[9] Sherwood R, Braud R, Bhattacharjee B. Slurpie: A cooperative bulk data transfer protocol. In: Proc. of the IEEE Infocom 2004. Hong Kong, 2004. 941−951. http://ieeexplore.ieee.org/search/selected.jsp?qry=%28slurpie%3A+a+cooperative+bulk+data+transfer+protocol%3Cin%3Emetadata%29&srch=1&resset=1331196&imageField.x=62&imageField.y=15&imageField=View+Selected+Items&chklist=1356981%40ieeecnfs

[10] Bit Torrent. 2004. http://bitconjurer.org/BitTorrent

[11] KaZaA. 2004. http://www.kazaa.com/

[12] Gnutella. 2004. http://www.gnutella.com/

[13] Saroiu S, Gummadi PK, Gribble SD. A measurement study of peer-to-peer file sharing systems. In: Proc. of the Multimedia Computing and Networking Conf. (MMCN) 2002. San Jose: ACM Press, 2002. 82. http://www.cs.washington.edu/homes/gribble/papers/mmcn.pdf

[14] Qiu DY, Srikant R. Modeling and performance analysis of bit torrent-like peer-to-peer networks. In: Proc. of the SIGCOMM 2004. Portland: ACM Press, 2004. 367−378. http://portal.acm.org/citation.cfm?id=1015508&coll=Portal&dl=ACM&CFID=67945259&CFTOKEN=25416258

[15] Ghemawat S, Gobioff H, Leung ST. The Google file system. In: Proc. of the 19th ACM SOSP 2003. Bolton Landing: ACM Press, 2003. 29−43. http://portal.acm.org/citation.cfm?id=945450&coll=Portal&dl=ACM&CFID=67945259&CFTOKEN=25416258

[16] Hua KA, Lee C, Hua CM. Dynamic load balancing in Multicomputer database systems using partition tuning. IEEE Trans. on Knowledge and Data Engineering, 1995,7(6):968−983.

[17] Aggarwal CC, Wolf JL, Yu PS. On optimal batching policies for video-on-demand storage server. In: Proc. of the IEEE ICMCS'96. Hiroshima: IEEE Computer Society, 1996. 253−258. http://ieeexplore.ieee.org/search/selected.jsp?qry=%28+on+optimal+batching policies+for+video-on-demand+storage+server%3Cin%3Emetadata%29&srch=1&resset=1331196&imageField.x=67&imageField.y=8&imageField=View+Selected+Items&chklist=534983%40ieeecnfs

陈明(1978－),男,广西玉林人,博士生,主要研究领域为对等网络,网格,分布式系统.

杨广文(1963－),男,博士,副教授,CCF 高级会员,主要研究领域为并行/高性能计算,网格.

王鼎兴(1937－),男,博士,教授,博士生导师,主要研究领域为并行/分布处理计算机系统.