一种混合式拓扑的 P2P VOD 系统设计①

王方雯 周景昊 (中国科学技术大学 计算机科学与技术学院 安徽 合肥 230027)

摘 要: 针对 P2P 视频点播系统的扩展性问题，从媒体源服务器、索引服务器两方面考虑，提出一种混合式拓扑结构的 P2P VOD 系统的设计模型。在 mesh 结构系统普遍采用的 gossip 拓扑层之上构建一层有结构的 P2P 网络拓扑，使得 DHT 可以用于拖动操作后的合作节点定位，以有效减轻索引服务器的压力。在该混合式拓扑的基础上进行多服务节点数据调度算法的设计，能很好地适应网络的异构性与节点的动态性，并有效降低媒体源服务器的负载，同时满足系统扩展性与视频服务质量的需求。

关键词: 视频点播；对等网络；混合拓扑；合作节点定位；多数据源调度

A Hybrid Topology Architecture for Peer-to-Peer Video-on-Demand System

WANG Fang-Wen, ZHOU Jing-Hao

(Department of Computer Science and Technology, University of Science and Technology of China, Hefei 230027, China)

Abstract: A hybrid topology architecture is presented to solve the problem of scalability in P2P VOD system. Based on the gossip topology of mesh-based system, this paper constructs a structured topology and provides a distributed collaborated nodes searching mechanism for dragging operation. Besides, the model designs an algorithm for data scheduling from multiple service nodes, which properly fits the heterogeneous and dynamic network environment, and also achieves good system scalability and high service quality.

Keywords: video-on-demand; peer-to-peer network; hybrid topology; collaborated nodes locating; multi-source data scheduling

1 引言

随着通信网络技术的迅速发展和宽带接入的普及化，用户的上网需求不断转变，流媒体视频服务日益成为最主要的网络业务之一。P2P 技术为实现大规模的视频服务提供了最为经济且有效的解决方式，目前 P2P 视频直播系统的研究已比较成熟并投入商业应用，而 P2P 点播系统中用户的异步点播需求及 VCR 请求给合作节点的定位带来了挑战[1]。

P2Cast[2]等树结构的 P2P VOD 系统采用完全集中管理的方式，完全依赖索引服务器进行节点加入时父节点的指派和父节点失效时的恢复，PeerVOD[3]、P2VOD[4]中则通过分布式的父节点失效恢复策略减轻组播树频繁重构给索引服务器带来的压力，但由于节点跳转的处理复杂，目前树结构的系统均不能很好的支持 VCR 操作。

网结构的 P2P VOD 系统，如 Gridcast 等[5]，节点在播放过程中维护一定的拓扑关系进行合作节点的选择与数据的调度。但在节点加入/跳转时，大部分系统采用完全依赖于索引服务器的方式进行拓扑关系的初始化，索引服务器需要维护所有节点的信息并提供查询响应，节点播放位置的实时刷新及频繁的拖动操作使其成为系统瓶颈。Vmesh[6]等系统中通过缓存固定视频段的方式引入 DHT[7]实现分布式的合作节点定位，但同时也引入了下载存储段的额外开销，且随机选择存储段的方式使得子段规模分布均匀不适应需求的动态变化。

本文提出一种混合式拓扑结构的 P2P VOD 系统的设计模型，基本思路如下：视频文件按照播放时间

① 收稿时间:2009-06-01

划分为多个长度相同的段，逻辑位置(文中指节点的调度位置)处于相同段的节点之间建立逻辑关系构成 gossip 层拓扑，节点间相互传播 gossip 信息保持高可靠性的连接。同时，节点对播放过程中获取的缓存段进行 DHT 注册，形成段种子拓扑层，使得 DHT 可以用于拖动操作后的合作节点定位，从而有效减轻索引服务器的压力，适用于大规模的系统部署。针对该混合式拓扑，文中进行了多服务节点间调度算法的设计，系统能很好的适应网络的异构性与节点的动态性并有效降低媒体源服务器的负载。

2　系统模型概述

混合结构的系统中包含三个部分：索引服务器(Tracker)、媒体源服务器(Source)、用户节点(peer)，用户节点缓存所有接收到的数据并依据缓存情况加入多层拓扑，如图 1 所示。首先给出系统模型中的相关定义。

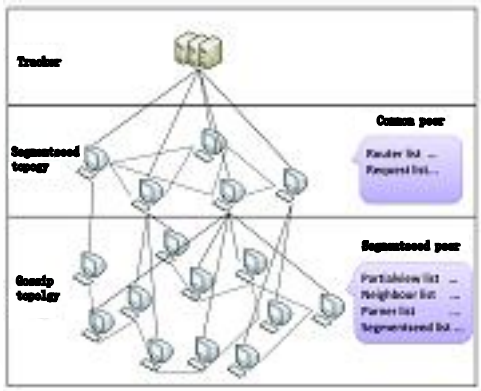

图 1　混合式拓扑系统结构

定义 1. 节点之间关于视频位置的距离称为节点间的逻辑距离，系统中采用满足播放需求的同时不断前向预取的数据下载策略，则调度窗口不断偏离播放位置，这里以调度窗口起始位置为节点的逻辑位置，记为 cur_schedule。逻辑位置是节点构建拓扑关系的依据。

定义 2. 采用分段策略，播放时长为 T 的视频文件被划分为大小相等的 S_{num} 段，视频段作为较大的划分粒度进一步被分割成若干数据块，数据块是网络中数据传输的基本单位。其中第 i 段时间范围为$((i-1)\times T/S_{num}, i\times T/S_{num})$，cur_shedule 位于同一视频段的节点称为同段节点。

定义 3. 节点与若干同段节点建立 UDP 连接，周期性地进行控制信息的交换，互为邻居关系。节点向若干同段节点发送数据请求进行数据块的传输，这些同段节点则称为合作节点。

定义 4. 如图 2 中所示，用户的频繁拖动操作以及合作节点的中途离开或网络抖动会造成节点缓存中的空缺，若节点的缓存区间内包含某段的所有数据块，则称其为该段的段种子节点。段种子节点之间形成结构化的拓扑网络。

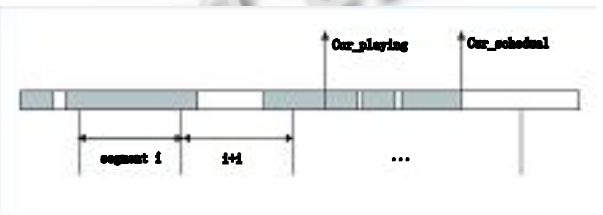

图 2　节点缓存示意

定义 5. 段种子节点响应若干逻辑位置居于该段的节点的数据请求，它们与合作节点并称为服务节点。

3　系统主要模块设计

3.1 Tracker 服务器设计

Tracker 服务器的作用是响应节点的查询请求并返回若干同段节点列表，传统的设计中采用维护系统所有节点信息的方式，无论在存储、查找和额外开销方面都耗费巨大，尤其在网络规模很大的情况下。而由于每次响应返回的节点数目有限，可采用仅维护部分节点最新信息的设计方式，如图 3 所示：

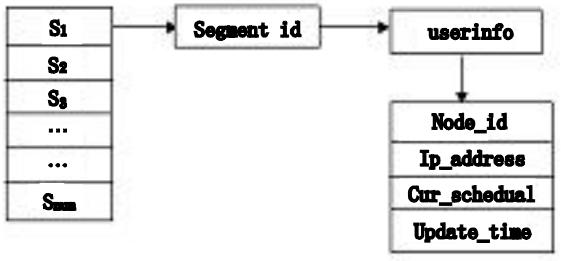

图 3　Tracker 服务器数据结构

Tracker 服务器为每个频道维护一个大小为 S_{num} 的数组，对于每个 segment_id 记录逻辑位置位于该段的若干播放节点的信息，节点条目的上限为 K，K 为稍大于返回节点列表的一个数。节点每隔一段时间

以一定概率 P 返回自己的逻辑位置信息到索引服务器，若段记录已满，则替换存在时间最长的条目，索引服务器中的段条目处于不断动态变化中，同时具备随机性和时间有效性。

假定系统中的节点数为 N,原刷新频率为 f，即信息条目的时间有效性为 $\Delta t=1/f$，上述设计方案可以在保证原时间有效性的基础上将刷新开销由 $n\times f$ 减小到 $S_{num}\times K\times f$,概率 P 的取值为：

$$P = S_{num} \times K / N \quad (1)$$

Tracker 服务器根据单位时间内到达的节点加入请求调节并发布该概率值，以 gossip 信息的形式在网络中传播，使得系统具有良好的自适应性。

3.2 分布式的查找机制

分析表明节点的拖动操作比加入操作更为频繁，是造成索引服务器压力的主要来源。采用分布式的跳转定位可消除拖动服务请求对索引服务器的完全依赖，本文通过构建有结构的段种子拓扑层来完成这一目标。

有结构的 DHT 网络在以文件为粒度的资源定位中发挥了高效的性能，我们通过将文件分段的方式将其引入合作节点定位机制。段种子节点以 movie_id、segment_id、mapped_coordinates 的组合作为标志 ID 注册加入该有结构的网络，即段种子拓扑层，节点同时缓存有多段内容时，会根据最近原则进行段 ID 的重新注册：

```
Segment ID choose{
        If (complete segment i downloading)
                Select segment i;
}
```

，节点位置具有相对稳定性。

采用 Kademlia 协议(简称 Kad)，与其他 DHT 实现技术比较，Kad 通过异或算法为距离度量基础，具有较高的路由查询速度。拓扑中的节点路由表由一些称之 K 桶的表格构造而成，每个节点保存有一些和自己距离范围在某区间内(2^i，i=0，1，..,n-1)的若干个节点信息，网络靠 UDP 协议交换信息。每个 K 桶覆盖距离的范围呈指数增长，使得距离自己近的节点信息多，离自己远的节点的信息少，路由查询过程可收敛。

拖动操作查询的目标是返回若干(假定为 a 个)目标位置的段种子节点，则需对 Kad 协议的远程操作 FIND_NODE 做相应修改，主要步骤如下：

① 以目标位置的段 ID 和节点网络坐标为查询目标 ID，计算目标 ID 与节点 ID 之间的距离 $d(x,y)=x\oplus y$；

② 从 x 的第$\lfloor \log d(x,y) \rfloor$个 K 桶中随机选取 a 个节点，若该桶中的记录数小于 a，则从后续多个桶中选择距离最接近 d(x,y)的 a 个节点，保证被选节点至少向目标 ID 推进 1 位，并向这 a 个节点发送 FIND_NODE 请求。

③ 接受到查询请求的每个节点，计算自己与目标 ID 之间的距离，重复过程 2 的选择操作，并将被选节点集返回至 x。

④ 重复对新返回的节点发送 FIND_NODE 请求操作，每次迭代之后距离至少缩小 1 位，直至返回节点集为空，因为目标 ID 实际上并不存在。

⑤ 从返回节点集中选择最接近目标 ID 的 a 个同段节点作为结果。

接收到 FIND_NODE 请求的节点根据发送者的信息更新路由表项，段种子节点向索引服务器注册完成加入拓扑网络的过程，索引服务器从路由表项中随机选取若干节点返回，新加入节点向这些节点发送以自身 ID 为目标的 FIND_NODE 查询,通过对邻近节点由远及近的逐步查询完成路由表项的初始化。

3.3 混合式拓扑系统中的点播服务流程

点播节点采用同时向若干同段节点与段种子节点请求数据服务的方式，下面详细介绍混合式拓扑的系统中其服务节点的获取流程。

① 加入节点向 Tracker 服务器发送点播请求,请求信息中包含 node_id、ip_address、cur_schedule、send_time。

② Tracker 服务器记录请求节点的信息，并返回若干 cur_schedule 位于同段的节点以及若干该段段种子节点的信息集合。

③ 请求节点与返回列表中的同段节点建立 UDP 连接的邻居关系，周期性的交换节点调度位置、段位图等信息。节点将收到的段位图与自身进行异或运算，取调度位置之后的“1”的位数为节点间的相关度表示，见公式(2)，选择相关度大的邻居作为合作节点。

$$Relativity_{B\to A}=\sum_{i=cur\_schedul \bmod n}^{n} chunkmaps[i]\oplus chunkmap[i] \quad (2)$$

④ 请求节点与返回列表中的段种子节点建立数据传输的 TCP 连接，段种子节点相应地维护 request_list 记录向其请求段数据的节点信息。

⑤ 节点间以 gossip 的方式传播邻居视图信息，更新并维护一定数量的邻居列表，数据传输率低于一定门限值的合作节点被淘汰并从邻居节点中重新选择。

⑥ 节点维护一定数量的段种子连接，由于段种子节点的路由表项设计为包含较多的同段、邻段节点信息，则当某一段种子节点离开网络或是重新注册时，可从其它段种子节点的路由表项中获取建立新连接的信息。

⑦ 节点执行拖动操作时，由正在连接的段种子发起目标段查询，并向返回节点请求它们的 request_list 作为候选邻居节点。

⑧ 点播用户收看完视频后离开网络，不需向其它节点发送通知，相关节点根据超时信息自动去除并替换列表中的条目信息。

3.4 多服务节点数据调度的策略

在完成合适的服务节点选择之后，节点间进行多对多的数据传输，调度策略主要包括数据块请求的指派与响应策略。

首先考虑数据块请求的指派，节点的数据来源包括三个部分：

① 媒体源服务器

节点尽可能地从服务节点获取数据的策略有利于点播系统的规模扩展，但为避免服务节点传输过程中的丢包或延迟造成的播放停顿，距离播放位置一定距离的区域被划分为紧急数据区域，这部分数据块下载直接向媒体源服务器请求。区域的大小为调度周期×视频播放率($\Delta T \times r$)。

② 合作节点

合作节点是数据下载的主要来源，为提供视频的正常播放，节点下载速率至少需要满足视频播放率。若存在 k 个合作节点时，每个合作节点需提供的带宽能力至少为 r/k。为有效利用合作节点的富余带宽能力，采用轮调度的方式，根据上一阶段的调度反馈调节下一阶段的请求量，如图 4 算法所示。这种块请求策略具有自适应性，并能充分利用网络中节点的富余带宽能力向前进行数据预取。

③ 段种子节点

段种子节点作为合作节点的补充，负责为紧急数据域之后、调度窗口前的数据块请求服务，避免随着播放过程的推进向源服务器请求这些数据，从而有效的分流服务器的负载。

服务节点将接收到的块请求按紧急程度(见公式(3))顺序插入请求队列，节点以最大上行带宽 upload(pi)响应每个请求，完成一个数据块的传输后即从队列首部取出下一个块请求并提供服务。节点优先响应紧急程度高的块请求，保证靠近播放点的数据被优先下载，段种子节点优先响应段数据请求，从而减小数据块请求因未被响应而进入紧急数据域的概率，有效减轻媒体源服务器负载。

$$Request\_urgency = cur\_request - cur\_playing \quad (3)$$

同时，节点设置超时门限，放弃在指定时间范围内未能响应的请求，从而满足调度的实时性。

```
ΔT：调度周期
cur_schedual：当前调度位置
W：调度窗口大小
Vinit=r×ΔT/N：节点性能初始值
Vcur_j：节点当前性能值
Vmax_j：节点最大性能值
ΔV：节点性能值的增量
chunkmap_j[i]:节点 j 的段位图的第 i 位
S：数据块大小
Void scheduler
    //确定本轮调度窗口大小
    For each partner j ： 1 to N
        If Vcur_j == Vmax_j
            Vcur_j = Vmax_j = Vmax_j + ΔV
        Else
            Vmax_j = min(r×ΔT /N,Vcur_j)
            ability_j = Vmax_j
        End
    W=∑Vcur_j,j∈partner list
    //进行数据块指派
    For each chunk i：cur_schedual to cur_schedual+W-1
        n = 0
        For j : 1 to N
            n = n+map_j[i]
            If n = 1 then
                k = argj{map_j[i] = 1}
                If(ability_k >= S)
                    supply(i) = k
                    ability_k=ability_k-S
                End
            Else
                dup_set[n]<-dup_set[n]∪ {i}
            End
        End
    End for i
    For n = 2 to N
        For each i ∈dup_set[n]
            k=argj{max{ability_j}}
            If(ability_k>=S)
                supply(i) = k
                ability_k=ability_k-S
            End
        End
    End
```

图 4　合作节点间数据调度算法

4　实验结果与性能分析

为验证系统性能，在 NS2 网络仿真环境下利用网络拓扑生成器 GT-ITM[8]生成基于中转-桩模型(transit-stub model)的网络拓扑作为实验的底层网

络。该网络拓扑共包含 1000 个随机产生的物理节点，其核心由 5 个平均拥有 4 个路由节点的相互连通的中转域构成，平均每个中转域与 10 个桩域相连，每个桩域平均包含 5 个物理节点。核心路由节点间的网络带宽为 1000Mbps，中转域路由节点间的网络带宽为 100Mbps，桩域路由节点间的网络带宽为 10Mbps，物理节点按接入带宽分为 4 种类型,如表 1 所示。

表 1 节点接入带宽分布

类型/带宽	ADSL_512（25%）	ADSL_1500（35%）	SDSL_512（25%）	DSL_2000（15%）
上行	128Kbit/s	256Kbit/s	512Kbit/s	2048Kbit/s
下行	512Kbit/s	1536Kbit/s	512Kbit/s	2048Kbit/s

媒体源服务器和索引服务器分别是一个核心路由节点，节目视频编码率为 300Kbps，播放时间为 90 分钟，每段时间为 5 分钟，数据块粒度为 1 秒钟的视频数据。初始缓冲为 10s，调度周期为 10s。

4.1 媒体源服务器负载

首先对系统中媒体源服务器的负载进行了测试，节点按照泊松分布加入网络，λ 取值为 0.5(number/second)，测试服务器负载情况并同理想调度模型及 Gridcast 模型进行比较。假定节点加入网络之后不执行拖动操作，节点有 50％的机率提前结束视频的播放并在视频播放完自动离开系统。

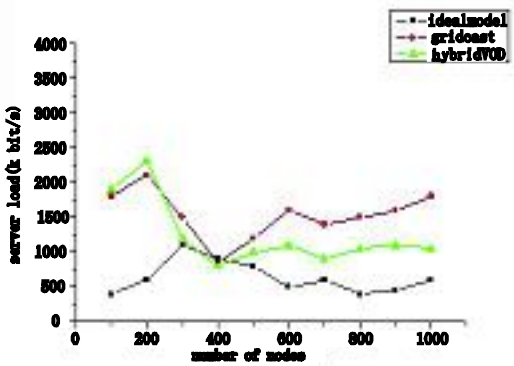

图 6 媒体源服务器负载对比图

图 6 的实验结果显示，随着节点规模的增大，源服务器负载维持在一个相对稳定的范围内，系统具有良好的扩展性。在某些特定时刻该负载值优于理想调度模型，这是因为理想调度模型并没有考虑数据的预取，节点根据播放速率下载视频并在播放结束时离开网络，靠近结束位置的节点只能从服务器端获取数据流，而系统中采用的前向预取调度策略使得这部分数据已经被分布在网络上。其次，当节点规模逐渐增大时，系统性能较 Gridcast 模型有所优势，这是因为系统中均匀分布的各段种子节点有效地分流了源服务器端的负载。

4.2 Tracker 服务器负载

接下来对存在拖动操作时索引服务器的压力进行测试。假定加入系统的节点有 90%进行拖动操作，其中 50%的节点执行 1 次跳转，20%的节点执行 2 次跳转，20%的节点执行 3 次跳转，跳转位置随机，记录不同强度 λ 下索引服务器的响应数。系统进入稳定阶段之后进行测试，测试结果见图 7。

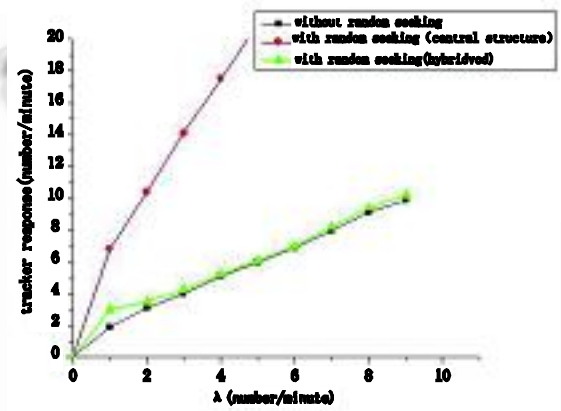

图 7 索引服务器压力对比图

当节点不执行拖动操作时，索引服务器仅响应节点的加入请求，其压力大小与节点的加入数呈线性递增关系。而在本文设计的混合式拓扑结构中，拖动操作对索引服务器几乎没有影响，仅当 λ 较小时距离标准曲线有一定偏差，这是由于拖动节点首先依赖段种子拓扑进行新邻居节点的查找，仅当节点保存的段种子索引信息失效时才会向索引服务器发送请求，网络规模较小时该失效比率较大。

5 总结

本文构造了一种混合式拓扑结构的 P2P VOD 系统模型，系统的主要特点是通过构造结构化的段种子拓扑层为点播用户提供分布式的 $O(\log N)$ 跳数内的拖动操作目标定位与邻居节点查找机制，同时段种子节点作为合作节点的补充可有效分流媒体源服务器的负载，能很好的适应动态性与异构性的网络环境。仿真实验表明该系统模型可有效降低媒体源服务器与索引服务器的负载，适用于大规模的视频点播服务。

(下转第 56 页)

(上接第 15 页)

参考文献

1 郑常熠,王新,赵进,薛向阳.P2P 视频点播内容分发策略.软件学报, 2007,18(11):2942－2954.

2 Guo Y, Suh K, Kurose J, Towsley D. P2Cast: P2P patching scheme for VoD service. Proc. of the WWW 2003. New York: ACM Press, 2003.301－309.

3 刘亚杰,窦文华.一种 P2P 环境下的 VoD 流媒体服务体系.软件学报, 2006,17(4):876－884.

4 Do T, Hua K, Tantaoui M. P2VoD: Providing fault tolerant video-on-demand streaming in peer-to-peer environment. Proc. of the IEEE ICC 2004. Paris: IEEE Communications Society, 2004.1467－1472.

5 Cheng B, Stein L, Jin H et al. GridCast: Improving peer sharing for P2P VoD. ACM Transactions on Multimedia Computing, Communications, and Applications, 2008, 4(4):47－77.

6 Ken WP, Jin YX, Chan S-HG. Distributed Storage to Support User Interactivity in Peer-to-Peer Video Streaming. Proc. of IEEE International Conference on Communications, June 2006.1:55－60.

7 Stoica I, Morris R, et al. Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications. IEEE/ACM Trans. Net, 2003,11(1):17－32.

8 Zegura E, Calvert K, Bhattachajee S. How to model an internetwork. Proc. IEEE, 1996.2:594－602.