基于 WebRTC 的对等视频会议模型

刘 浩,李千目

(南京理工大学计算机科学与工程学院,江苏 南京 210094)

[**摘要**] 传统对等视频会议系统对网络和硬件条件要求较高,容易导致视频会议交互质量较差. 针对这个问题,本文提出一种适用于单一时刻关注目标数小于参与者总数时(如移动端会议)的对等视频会议模型(WebRTC based video conference model,WVCM). 该模型基于 WebRTC 媒体处理引擎,减少同时关注的目标数,使用少于传统模型的资源消耗就达到了与传统模型相当的规模,并通过目标加入和退出算法实现对会议中所关注目标的快速切换以及模型的维护. 实验结果表明,WVCM 能在保证会议质量的前提下有效减少网络流量并扩大会议规模.

[**关键词**] 视频会议,WebRTC,对等网络

[**中图分类号**]TP39 [**文献标志码**]A [**文章编号**]1672-1292(2014)04-0045-06

Peer-to-Peer Video Conference Model Based on WebRTC

Liu Hao,Li Qianmu

(School of Computer Science and Engineering,Nanjing University of Science and Technology,Nanjing 210094,China)

Abstract:Traditional peer-to-peer video conference system has very high requirements for both network and hardware condition,which causes the quality of video conference to degrade. To improve this situation,a WebRTC Based Video Conference model,WVCM,is proposed in this paper. This model can reach the same scale as traditional conference model with lower resource cost by reducing the amount of targets observed at the same time. Then,a target joining and leaving algorithm is introduced to enable quick target switch. These features make WVCM very suitable for the situation,such as mobile conference,where conference participants are more than targets to be watched simultaneously. Experiments show that WVCM can reduce traffic and scale up the conference without harming service quality.

Key words:video conference,WebRTC,peer-to-peer network

视频会议对于人们来说早已不再陌生,它可以让多人同时进行远距离实时视频、音频交流. WebRTC(Web real-time communication)是 Google 的一个开源项目,它的出现可以帮助 Web 浏览器、个人电脑或手机实现实时多媒体通讯功能. WebRTC 提供了各种实现视频会议的核心技术,包括音视频的采集、编解码、信号优化和处理以及网络传输等[1,2].

传统的视频会议系统往往采用 C/S 或 B/S 架构,媒体流通过服务器转发给各个节点. 但是这种模式对中央服务器要求较高,容易发生故障,且随着用户数量的增加容易出现网络传输瓶颈. P2P 网络的不断发展使得新的视频会议架构方案成为可能[3,4]. 基于 P2P 的视频会议模型主要有以下几种:

基于全连接的模型. 这种结构的特点是会议节点与其他每个节点都建立一个连接,设计简单,支持多方全交互会议. 但是随着节点数增加,节点负担会急剧上升,使得这种模型的会议规模不会很大,如美国康奈尔大学的 CU-SeeMe[5] 视频会议系统,只支持 4 个用户同时在线工作.

树分发模型. 如高昂等提出的 VCStream[6],其特点是通过建立组播树,将数据由父节点到子节点由上而下地分发,从而降低了对网络及设备的依赖,但是只对单源会议模式提供良好支持.

收稿日期:2014-07-20.

基金项目:国家自然科学基金(61272419)、江苏省自然科学基金(BK2011370)、中国博士后基金(2012M521089)、江苏省博士后资助计划(1201044C)、江苏省产学研联合创新基金-前瞻性联合研究项目(BY2012022).

通讯联系人:李千目,教授,博士生导师,研究方向:信息安全、物联网应用系统、大数据处理系统. E-mail:liqianmu@126.com

在有些场景,参与到一次多方全交互视频会议中的用户并不需要同时关注其他所有用户. 例如,一次会议中,仅有少数关键人物在会议上作报告,则其他参与者平时只需关注他们,在需要的时候对关注目标进行切换即可;或者,用户参与移动端视频会议,受屏幕尺寸限制,不能同时关注过多目标,需要不时对关注目标进行选择. 针对上述情况,本文提出基于 WebRTC 的视频会议模型(WebRTC based video conference model,WVCM).

1　WVCM 模型原理

WVCM 模型结合了全连接模型和树分发模型的优点,通过减少同时关注的目标个数并提供目标切换功能,实现仿全交互的会议模型. 模型拓扑结构如图 1 所示,其中实线代表由节点 A 产生的媒体流,虚线代表由 B 产生的媒体流. 图 1 中,节点 C、D、E、F、G 都在请求来自 A 的媒体流,其中 C、F 直接从 A 处获取媒体流,而 D、E、G 则是以间接的方式从 C 或 F 处获取来自 A 的媒体流. 节点 B 的情况与之类似.

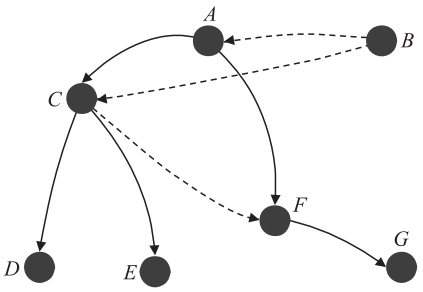

图 1　WCVM 拓扑结构

Fig. 1　WCVM's topology

WVCM 的主要工作流程包括初始化,请求连接,维护连接和数据传输等,其中请求连接和维护连接两部分将在下文给出具体介绍. WVCM 的整体流程如图 2 所示.

WVCM 的设计主要包括两部分. 第一部分是模型的流量分发算法的设计,包括模型中角色的划分及指定,媒体源的选择算法等;第二部分则主要研究了如何将流量分发算法与 WebRTC 集成.

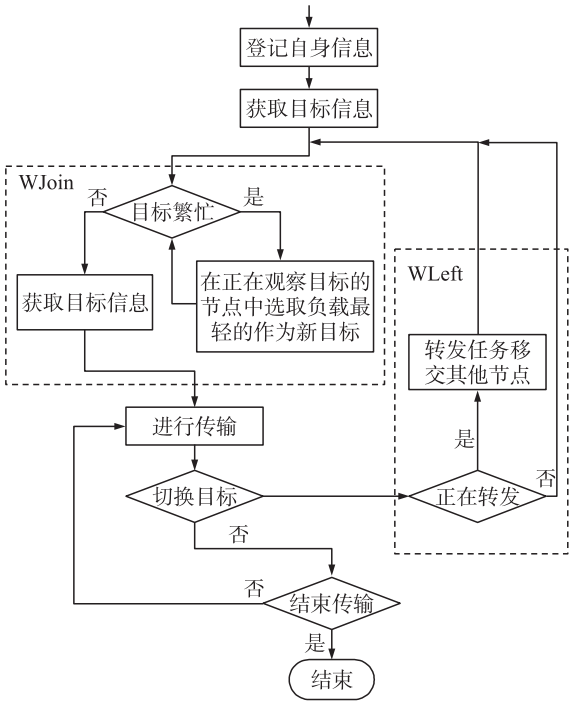

图 2　WVCM 流程

Fig. 2　Flow chart of WVCM

2　流量分发算法的设计

2.1　模型角色的划分及指定

参与到流量分发算法中的所有节点都被指派为模型对等体(WVCM peer,WPeer),主要任务为流发送、接收和转发. 在每一次会议中,发起会议的 WPeer 还会被指派一个额外角色,即会议发起者(WVCM initiator,WInitiator),负责协助各节点在会议发起阶段进行初始化. 这里要说明一点,一个 WInitiator 同时也是一个 WPeer,具体实现上采用面向对象思想,即 WInitiator 是 WPeer 的继承类.

称每 1 个 WPeer 为 1 个节点,它是 1 个五元组 (R,S,T,m,n),其中 R 为所有会议中节点的集合,S 为请求由该 WPeer 产生的媒体流的节点的集合,T 为三元组 (a,b,c) 的集合,其中 a 是直接与该 WPeer 建立媒体连接的节点,b 为当前媒体的产生节点,c 为自然数,每个转发连接对应 1 个集合 S 和 T,m 为该节点为每条媒体流进行转发的传输连接数阈值,n 为该节点正在播放的媒体流数量阈值.

2.2　会议发起阶段

会议的发起阶段由 WInitiator 主导,由它接收所有节点的参会请求,搜集它们的 ID 及地址等信息,并连同其自身信息存至集合 A. 此外,WInitiator 还负责响应其他节点要求同步其集合 R 的请求,具体流程如图 3 所示.

2.3　会议媒体流连接的动态管理

在会议成功发起后,每个 WPeer 的集合 N 都已得到同步. 此时,节点首先为响应来自其他节点的媒体流连接获取请求做准备,然后就开始向其目标发起获取媒体流的请求,具体流程如下:

Procedure:RequestStream
Input:s 为请求者,t 为被请求者
1)BEGIN
2)　令 s.S,s.T 为空集
3)　start_listen(s)
4)　cnn=req_trans_conn(s,t)
5)　s 通过 cnn 获取 t 产生的媒体流
6)END

其中,req_trans_conn(s,t)将阻塞请求发起方 s,直至 t 返回可用连接.

节点在接收到 req_trans_conn 请求后,将运行算法 WJoin. 该算法根据节点自身运行情况决定由自己或已存在的请求节点为当前请求者提供媒体流传输连接. 具体过程如下:

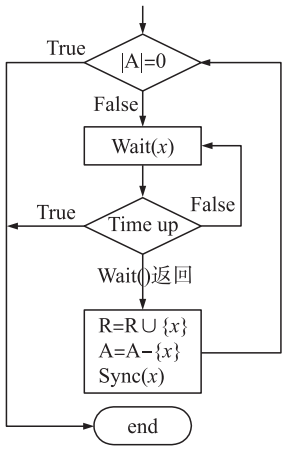

图 3　WInitiator 会议发起流程

Fig. 3　Initialization with WInitiator

Algorithm:WJoin
Input:(p,q,t),p 为请求者,q 为数据源,t 为转发者
Output:o 为媒体的实际提供者
1)　if(q==t){
2)　　q.S=q.S∪{p}
3)　　if(q.|T|<q.m){
4)　　　q.T=q.T∪{(p,q,0)}
5)　　　return q
6)　　}else{
7)　　令 x=(a,b,c)是 q.T 中 c 值最小的元素
8)　　以 x=(a,b,c+1)更新 q.T
9)　　return WJoin(p,q,x)
10)}else{
11)　　if(t.|T|<t.m){
12)　　　t.T=t.T∪{(p,q,0)}
13)　　　return t
14)　}else{
15)　　令 x=(a,b,c)是 t.T 中 c 值最小的元素
16)　　以 x=(a,b,c+1)更新 t.T
17)　　return WJoin(p,q,t)
18)}}

当节点已请求媒体流个数到达 n 时(即观看目标的个数已达上限),此时需要用新的连接替换已有连接. 节点调用 WLeft 算法,在保证整个模型的传输不被破坏的前提下,断开与已有目标的连接,从而为选择新目标做准备,算法 WLeft 具体如下:

Algorithm:WLeft
Input:(p,q,t),t 转发者,q 为数据源,p 为请求者
Output:新的模型,p 不再请求 q 产生的媒体流
1)if(p.T==空集){
2)　　t.T=t.T-{p}
3)}else{
4)　　设 s 为最后一个加入 q.S 的元素
5)　　令 s 从 t 接收来自 q 的数据

```
6)    for u in p.T{
7)      令 u 从 s 中接收来自 q 的数据
8)    }
9)    q.S=q.S-{s}
10)}
```

3　集成 WebRTC 媒体处理引擎

3.1　WebRTC 媒体处理引擎的架构

WebRTC 的媒体处理引擎的架构遵循分层的思想,语音引擎与视频引擎相互独立,并且仅涉及数字信号的处理. 良好的接口设计使得媒体流的采集和渲染以及媒体流的传输即可使用 WebRTC 自带的模块,也可以较容易地扩展第三方模块. WebRTC 媒体处理引擎的架构层次如图 4 所示.

WebRTC 媒体流引擎主要由 3 大部分组成,即语音引擎、视频引擎和数据传输模块. 其中语音引擎负责采集语音信号,对语音数字信号进行处理(如降噪、消除回声等),对语音信号进行编码、解码以及对语音信号进行渲染和播放. 语音引擎集成了目前大部分主流的 Codec,如 G711、iLBC 和 iSAC 等,能较好地满足主流业务的需求. 另外,如果用户需要,还可以利用 WebRTC 语音引擎的接口自行集成其他 Codec.

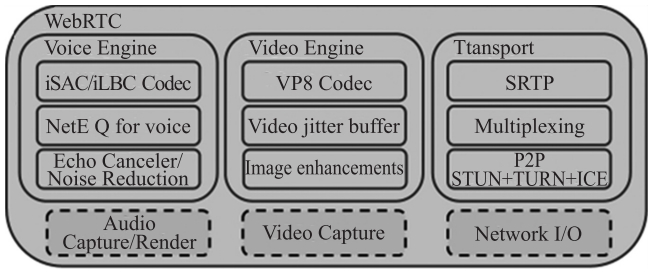

图 4　WebRTC 媒体引擎架构

Fig. 4　WebRTC's media engine

视频引擎负责对视频信号进行采集、处理以及渲染. 目前该视频引擎支持的视频 Codec 只有 VP8,它是一个开放的图像压缩格式,最早由 On2 Technologies 开发,随后由 Google 发布. VP8 拥有较好的效率,但应用不够广泛,也因此很少有硬件厂商提供针对 VP8 的硬件加速功能. 幸运的是,用户可以自行将 H264 集成到 WebRTC 的视频引擎框架中,使用来自设备的硬件加速功能,进一步提高编解码效率.

Transport 模块实际上是 WebRTC 定义的一套接口,用来为上层提供底层的数据传输功能. 虽然 WebRTC 针对 Transport 模块给出的建议是由用户自己实现与业务相关的功能,但它也给出了一个实验性质的实现,可以完成简单的演示.

3.2　集成方案

WebRTC 的语音引擎(voice engine)与视频引擎(video engine)对数据的获取与发送都依赖于 Transport 模块:由 Transport 模块发送已编码并打包的数据包,并由 Transport 模块负责接收此类数据包并交由上层进行处理. 本文通过注册外部 Transport 的方法来完成媒体流分发算法与 WebRTC 的集成. Transport 模块对外主要体现为两个接口, UdpTransport 和 UdpTransportData,如图 5 所示.

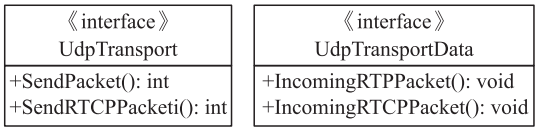

图 5　Transport 模块接口定义

Fig. 5　Definition of transport module

媒体流分发算法与 WebRTC 的集成具体可以参照 WebRTC 原有 Transport 模块的做法,即使用 socket 建立 UDP 传输,如图 6 所示.

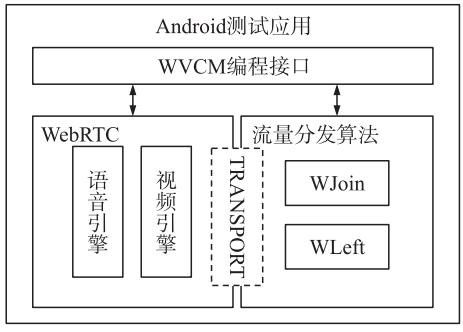

图 6　WVCM 系统集成模块图

Fig. 6　Overview of WVCM modules

原有模块提供了 P2P 多媒体通讯的功能,本文采用上述流量分发算法改写原有模块,建立独立的监听 socket 来辅助过程 RequestStream 处理收到的请求. 通过 WJoin 算法来建立传输媒体流所需的 socket,并把对应的地址及端口发送给请求者,完成传输连接的建

立. 在节点切换目标时,调用 WLeft 算法对传输结构进行维护,再通过 WJoin 算法获取新目标. 在 WLeft 过程中,已建立的 socket 可以复用,因为节点是通过端口号来区分媒体流进而防止多个媒体流交错在一起. 这样,切换目标的开销被大大降低. 另外,节点在本地缓冲收到的 UDP 报文,并在向上层提交报文的同时,转发给目标节点,从而完成流量分发.

4　实验结果与分析

4.1　实验环境

本文实验在 ubuntu-12.04.2-server-amd64 下完成外部 Transport 和 WebRTC 媒体流引擎的构建,使用 NDK-r8b 结合 jni 层构建共享库 libwvcm.so,并使用该共享库替代 WebRTC 提供的媒体引擎 Demo 程序原有的共享库来完成测试程序的开发. 测试程序的运行平台是搭载 Android2.3 系统的智能手机(处理器,高通 Krait 400,2.2 GHz;内存 2 GB),视频分辨率为 320 * 240,网络环境为 54 Mbps 的 WLAN.

4.2　实验过程与结果

本文首先建立基于 P2P 的 8 路传统对等视频会议,监测各个节点网络流量状况. 然后利用 WVCM 同样发起 8 路对等视频会议,并分别对关键参数 m、n 设置不同数值. 两种模型的比较结果如图 7 所示. 从图中可以看出,m 的取值对全局平均流量基本没有影响,但 m 的增大会明显加重模型中最繁忙节点的负荷.

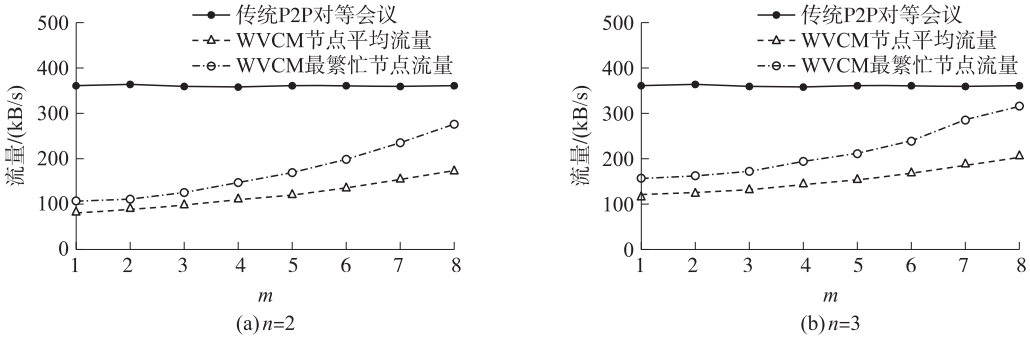

图 7　流量比较结果

Fig. 7　Comparison of bit rate

其次本文建立基于 P2P 的传统对等视频会议,监测视频帧速 FPS. 然后用 WVCM 建立对等视频会议,令 n 取不同应值,监测其 FPS 并与前者比较,结果如图 8 所示. 可以看出,与传统 P2P 模型相比,同一时刻收看相同路数的媒体,WVCM 在扩大了会议规模的同时也保证了视频会议的质量未受明显影响.

最后本文就切换会议目标的延迟在不同环境下进行测试,网络延迟利用 FreeBSD6.1 进行模拟. 测试结果见表 1. 可以看出,m 和网络延迟的增大会造成切换延迟的增加,n 对切换延迟无明显影响.

表 1　目标切换延迟

Table 1　Delay of target switch

m	n	网络延迟/ms	平均切换延迟/s
3	2	30	1.5
3	2	100	2.4
3	3	30	1.5
3	3	100	2.4
4	2	30	1.7
4	2	100	3.1
4	3	30	1.7
4	3	100	3.1

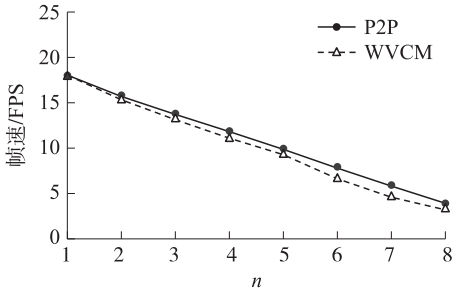

图 8　帧速比较结果

Fig. 8　Comparison of FPS

5　结语

本文借鉴传统 P2P 对等模型以及主从式模型,并结合 WebRTC 提出了 WVCM 对等视频会议模型. WVCM 摒弃同时播放所有对等节点媒体的模式,由用户选择某一时刻只播放若干个节点媒体,从而能有效降低节点负荷,防止某节点负荷过重崩溃,并在保证会议表现力不受明显影响的前提下,扩大同等外部

条件下的会议规模. 在下一步工作中,将通过 WebRTC 媒体处理引擎的编码器集成硬件加速功能,改善会议表现力,提高会议帧速,并进一步优化 WJoin 算法,减少转发链的长度,从而降低切换目标时的延迟.

[参考文献](References)

[1] Google Chrome Team. WebRTC General Overview and References[EB/OL]. [2014-04-01] http://www.webrtc.org/reference/architecture.

[2] 胡平,聂朋朋,陆建德. 典型 P2P 流媒体模型及其关键技术[J]. 计算机工程,2009,35(3):60-62.
Hu Ping, Nie Pengpeng, Lu Jiande. Typical P2P media stream model and related key techniques[J]. Computer Engineering, 2009,35(3):60-62. (in Chinese)

[3] Zhe Xiang, Qian Zhang. Peer-to-peer based multimedia distribution service[J]. IEEE Transactions on Multimedia, 2004, 6(2):343-355.

[4] Tran D A, Hua K A, Do T. Zigzag: An efficient peer-to-peer scheme for media streaming[C]//Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. San Francisco: IEEE, 2003, 2: 1 283-1 292.

[5] Jefferson Han, Brian Smith. CU-SeeMe VR immersive desktop teleconferencing[J]. ACM Multimedia, 1996, 19(8): 199-207.

[6] Gao Ang, Xu Shuang, Li Zengzhi, et al. P2P-based media streaming distribution model and its application in video conference system[J]. Journal of Southwest Jiaotong University, 2013, 48(1): 88-93.

[责任编辑:丁 蓉]