转

H.264 NAL层解析

1．引言

H.264的主要目标：

1．高的视频压缩比

2．良好的网络亲和性

解决方案：

VCL video coding layer 视频编码层

NAL network abstraction layer 网络提取层

VCL：核心算法引擎，块，宏块及片的语法级别的定义

NAL：片级以上的语法级别（如序列参数集和图像参数集），同时支持以下功能：独立片解码，起始码唯一保证，SEI以及流格式编码数据传送

VCL设计目标：尽可能地独立于网络的情况下进行高效的编解码

NAL设计目标：根据不同的网络把数据打包成相应的格式，将VCL产生的比特字符串适配到各种各样的网络和多元环境中。

NALU头结构：NALU类型(5bit)、重要性指示位(2bit)、禁止位(1bit)。

NALU类型：1～12由H.264使用，24～31由H.264以外的应用使用。

重要性指示：标志该NAL单元用于重建时的重要性，值越大，越重要。

禁止位：网络发现NAL单元有比特错误时可设置该比特为1，以便接收方丢掉该单元。

2．NAL语法语义

NAL层句法：

在编码器输出的码流中，数据的基本单元是句法元素。

句法表征句法元素的组织结构。

语义阐述句法元素的具体含义。

分组都有头部，解码器可以很方便的检测出NAL的分界，依次取出NAL进行解码。

但为了节省码流，H.264没有另外在NAL的头部设立表示起始位置的句法元素。

如果编码数据是存储在介质上的，由于**NAL是依次紧密相连的，解码器就无法在数据流中分辨出每个NAL的起始位置和终止位置**。

解决方案：在每个NAL前添加起始码：0X000001

在某些类型的介质上，**为了寻址的方便，要求数据流在长度上对齐，或某个常数的整数倍。所以在起始码前添加若干字节的0来填充**。

检测**NAL**的开始：

0X000001和0X00000001

我们必须考虑当**NAL**内部出现了**0X000001**和**0X000000**

解决方案：

H.264提出了"防止竞争"机制：

0X000000——0X00000300

0X000001——0X00000301

0X000002——0X00000302

0X000003——0X00000303

为此，我们可以知道：

在NAL单元中，下面的三字节序列不应在任何字节对齐的位置出现

0X000000

0X000001

0X000002

Forbidden_zero_bit =0;

Nal_ref_idc：表示**NAL**的优先级。**0～3**，取值越大，表示当前**NAL**越重要，需要优先受到保护。如果当前**NAL**是属于参考帧的片，或是序列参数集，或是图像参数集这些重要的单位时，本句法元素必需大于**0**。

Nal_unit_type：当前**NAL** 单元的类型

3．H.264的NAL层处理

结构示意图：

NAL以**NALU**（**NAL unit**）为单元来支持编码数据在基于分组交换技术网络中传输。

它定义了符合传输层或存储介质要求的数据格式，同时给出头信息，从而提供了视频编码和外部世界的接口。

NALU：定义了可用于基于分组和基于比特流系统的基本格式

RTP封装：只针对基于**NAL**单元的本地**NAL**接口。

三种不同的数据形式：

SODB 数据比特串－－>最原始的编码数据 （**raw**）

RBSP 原始字节序列载荷－－>在**SODB**的后面填加了结尾比特（**RBSP trailing bits**一个**bit"1"**）若干比特**"0",**以便字节对齐

EBSP 扩展字节序列载荷**-->**在**RBSP**基础上填加了仿校验字节（**0X03**）它的原因是： 在**NALU**加到**Annexb**上时，需要添加每组**NALU**之前的开始码**StartCodePrefix,**如果该**NALU**对应的**slice**为一帧的开始则用**4**位字节表示，**ox00000001,**否则用**3**位字节表示**ox000001.**为了使**NALU**主体中不包括与开始码相冲突的，在编码时，每遇到两个字节连续为**0**，就插入一个字节的**0x03**。解码时将**0x03**去掉。也称为脱壳操作

处理过程：

1． 将VCL层输出的SODB封装成nal_unit， **Nal_unit**是一个通用封装格式，可以适用于有序字节流方式和**IP**包交换方式。

2． 针对不同的传送网络（电路交换|包交换），将nal_unit 封装成针对不同网络的封装格 式。

第一步的具体过程：

VCL层输出的比特流SODB（String Of Data Bits），到nal_unit之间，经过了以下三步处理：

1.SODB字节对齐处理后封装成RBSP（Raw Byte Sequence Payload）。

2.为防止RBSP的字节流与有序字节流传送方式下的SCP（start_code_prefix_one_3bytes，0x000001）出现字节竞争情形，循环检测RBSP前三个字节，在出现字节竞争时在第三字节前加入**emulation_prevention_three_byte** （**0x03**）

具体方法：

```
nal_unit( NumBytesInNALunit ) {

forbidden_zero_bit

nal_ref_idc

nal_unit_type

NumBytesInRBSP = 0

for( i = 1; i < NumBytesInNALunit; i++ ) {
```

if(i + 2 < NumBytesInNALunit && next_bits(24) = = 0x000003) {

rbsp_byte[NumBytesInRBSP++]

rbsp_byte[NumBytesInRBSP++]

i += 2

emulation_prevention_three_byte /* equal to 0x03 */

} else

rbsp_byte[NumBytesInRBSP++]

}

}

3. 防字节竞争处理后的RBSP再加一个字节的header(forbidden_zero_bit+ nal_ref_idc+ nal_unit_type)，封装成nal_unit.

第二步的具体过程：

case1：有序字节流的封装

byte_stream_nal_unit(NumBytesInNALunit) {

while(next_bits(24) != 0x000001)

zero_byte /* equal to 0x00 */

if(more_data_in_byte_stream( )) {

start_code_prefix_one_3bytes /* equal to 0x000001 */ nal_unit(NumBytesInNALunit)

}

}

类似H.320和MPEG-2/H.222.0等传输系统，**传输NAL作为有序连续字节或比特流，同时要依靠数据本身识别NAL单元边界。在这样的应用系统中，H.264/AVC规范定义了字节流格式，每个NAL单元前面增加3个字节的前缀，即同步字节。**在比特流应用中，**每个图像需要增加一个附加字节作为边界定位。**还有一种可选特性，在字节流中增加附加数据，用做扩充发送数据量，能实现快速边界定位，恢复同步

Case2：**IP网络的RTP打包封装**

分组打包的规则

(1)额外开销要少，使MTU尺寸在100～64k字节范围都可以；

(2)不用对分组内的数据解码就可以判别该分组的重要性；

(3)载荷规范应当保证不用解码就可识别由于其他的比特丢失而造成的分组不可解码；

(4)支持将**NALU**分割成多个**RTP**分组；

(5)支持将多个**NALU**汇集在一个**RTP**分组中。

RTP的头标可以是NALU的头标，并可以实现以上的打包规则。

一个RTP分组里放入一个NALU，将NALU(包括同时作为载荷头标的NALU头)放入RTP的载荷中，设置RTP头标值。为了避免IP层对大分组的再一次分割，片分组的大小一般都要小于MTU尺寸。由于包传送的路径不同，解码端要重新对片分组排序，RTP包含的次序信息可以用来解决这一问题。

NALU分割

对于预先已经编码的内容，NALU可能大于MTU尺寸的限制。虽然IP层的分割可以使数据块小于64千字节，但无法在应用层实现保护，从而降低了非等重保护方案的效果。由于UDP数据包小于64千字节，而且一个片的长度对某些应用场合来说太小，所以应用层打包是RTP打包方案的一部分。

新的讨论方案(IETF)应当符合以下特征：

(1)NALU的分块以按RTP次序号升序传输；

(2)能够标记第一个和最后一个NALU分块；

(3)可以检测丢失的分块。

NALU合并

一些NALU如SEI、参数集等非常小，将它们合并在一起有利于减少头标开销。已有两种集合分组：

(1)单一时间集合分组(STAP)，按时间戳进行组合；

(2)多时间集合分组(MTAP)，不同时间戳也可以组合。

NAL规范视频数据的格式，主要是提供头部信息，以适合各种媒体的传输和存储。NAL支持各种网络，包括：

1．任何使用RTP/IP协议的实时有线和无线Internet 服务

2．作为MP4文件存储和多媒体信息文件服务

3．MPEG-2系统

4．其它网

NAL规定一种通用的格式，既适合面向包传输，也适合流传送。实际上，包传输和流传输的方式是相同的，不同之处是传输前面增加了一个起始码前缀

在类似Internet/RTP面向包传送协议系统中，包结构中包含包边界识别字节，在这种情况下，不需要同步字节。

NAL单元分为VCL和非VCL两种

VCL NAL单元包含视频图像采样信息，

非VCL包含各种有关的附加信息，例如参数集（头部信息，应用到大量的VCL NAL单元）、提高性能的附加信息、定时信息等

参数集：

参数集是很少变化的信息，用于大量VCL NAL单元的解码，分为两种类型：

1．序列参数集，作用于一串连续的视频图像，即视频序列。

两个IDR图像之间为序列参数集。IDR和I帧的区别见下面。

2． 图像参数集，作用于视频序列中的一个或多个个别的图像序列和图像参数集机制，减少了重复参数的传送，每个VCL NAL单元包含一个标识，指向有关的图像参数集，每个图像参数集包含一个标识，指向有关的序列参数集的内容因此，只用少数的指针信息，引用大量的参数，大大减少每个VCL NAL单元重复传送的信息。

序列和图像参数集可以在发送VCL NAL单元以前发送，并且重复传送，大大提高纠错能力。序列和图像参数集可以在"带内"，也可以用更为可靠的其他"带外"通道传送。

存储单元：

一组指定格式的NAL单元称为存储单元，每个存储单元对应一个图像。每个存储单元包含一组VCL NAL单元，组成一个主编码图像，VCL NAL单元由表示视频图像采样的像条所组成。存储单元前面可以加一个前缀，分界存储单元，附加增强信息（SEI）（如图像定时信息）也可以放在主编码图像的前面。主编码图像后附加的VCL NAL单元，包含同一图像的冗余表示，称为冗余编码图像，当主编码图像数据丢失或损坏时，可用冗余编码图像解码。

编码视频序列

一个编码视频序列由一串连续的存储单元组成，使用同一序列参数集。每个视频序列可独立解码。编码序列的开始是即时刷新存储单元（IDR）。IDR是一个I帧图像，表示后面的图像不用参考以前的图像。一个NAL单元流可包含一个或更多的编码视频序列。

RTP协议：

实时传输协议（Real-time Transport Protocol，RTP）是在Internet上处理多媒体数据流的一种网络协议，利用它能够在一对一（单播）或者一对多（multicast，多播）的网络环境中实现传流媒体数据的实时传输。RTP通常使用UDP来进行多媒体数据的传输，但如果需要的话可以使用TCP或者ATM等其它协议，整个RTP协议由两个密切相关的部分组成：RTP数据协议和RTP控制协议。实时流协议（Real Time Streaming Protocol， RTSP）最早由Real Networks和Netscape公司共同提出，它位于RTP和RTCP之上，其目的是希望通过IP网络有效地传输多媒体数据。

RTP数据协议

RTP数据协议负责对流媒体数据进行封包并实现媒体流的实时传输，每一个RTP数据报都由头部（Header）和负载（Payload）两个部分组成，其中头部前12个字节的含义是固定的，而负载则可以是音频或者视频数据。RTP数据报的头部格式如图1所示：

其中比较重要的几个域及其意义如下：

CSRC记数（CC） 表示CSRC标识的数目。CSRC标识紧跟在RTP固定头部之后，用来表示RTP数据报的来源，RTP协议允许在同一个会话中存在多个数据源，它们可以通过RTP混合器合并为一个数据源。例如，可以产生一个CSRC列表来表示一个电话会议，该会议通过一个RTP混合器将所有讲话者的语音数据组合为一个RTP数据源。

负载类型（PT） 标明RTP负载的格式，包括所采用的编码算法、采样频率、承载通道等。例如，类型2表明该RTP数据包中承载的是用ITU G.721算法编码的语音数据，采样频率为8000Hz，并且采用单声道。

序列号 用来为接收方提供探测数据丢失的方法，但如何处理丢失的数据则是应用程序自己的事情，RTP协议本身并不负责数据的重传。

时间戳 记录了负载中第一个字节的采样时间，接收方能够时间戳能够确定数据的到达是否受到了延迟抖动的影响，但具体如何来补偿延迟抖动则是应用程序自己的事情。从RTP数据报的格式不难看出，它包含了传输媒体的类型、格式、序列号、时间戳以及是否有附加数据等信息，这些都为实时的流媒体传输提供了相应的基础。RTP协议的目的是提供实时数据（如交互式的音频和视频）的端到端传输服务，因此在RTP中没有连接的概念，它可以建立在底层的面向连接或面向非连接的传输协议之上；RTP也不依赖于特别的网络地址格式，而仅仅只需要底层传输协

议支持组帧（Framing）和分段（Segmentation）就足够了；另外RTP本身还不提供任何可靠性机制，这些都要由传输协议或者应用程序自己来保证。在典型的应用场合下，RTP一般是在传输协议之上作为应用程序的一部分加以实现的，如图2所示：

RTCP控制协议

RTCP控制协议需要与RTP数据协议一起配合使用，当应用程序启动一个RTP会话时将同时占用两个端口，分别供RTP和RTCP使用。RTP本身并不能为按序传输数据包提供可靠的保证，也不提供流量控制和拥塞控制，这些都由RTCP来负责完成。通常RTCP会采用与RTP相同的分发机制，向会话中的所有成员周期性地发送控制信息，应用程序通过接收这些数据，从中获取会话参与者的相关资料，以及网络状况、分组丢失概率等反馈信息，从而能够对服务质量进行控制或者对网络状况进行诊断。

RTCP协议的功能是通过不同的RTCP数据报来实现的，主要有如下几种类型：

SR 发送端报告，所谓发送端是指发出RTP数据报的应用程序或者终端，发送端同时也可以是接收端。

RR 接收端报告，所谓接收端是指仅接收但不发送RTP数据报的应用程序或者终端。

SDES 源描述，主要功能是作为会话成员有关标识信息的载体，如用户名、邮件地址、电话号码等，此外还具有向会话成员传达会话控制信息的功能。

BYE 通知离开，主要功能是指示某一个或者几个源不再有效，即通知会话中的其他成员自己将退出会话。

APP 由应用程序自己定义，解决了RTCP的扩展性问题，并且为协议的实现者提供了很大的灵活性。

RTCP数据报携带有服务质量监控的必要信息，能够对服务质量进行动态的调整，并能够对网络拥塞进行有效的控制。由于RTCP数据报采用的是多播方式，因此会话中的所有成员都可以通过RTCP数据报返回的控制信息，来了解其他参与者的当前情况。

在一个典型的应用场合下，发送媒体流的应用程序将周期性地产生发送端报告SR，该RTCP数据报含有不同媒体流间的同步信息，以及已经发送的数据报和字节的计数，接收端根据这些信息可以估计出实际的数据传输速率。另一方面，接收端会向所有已知的发送端发送接收端报告RR，该RTCP数据报含有已接收数据报的最大序列号、丢失的数据报数目、延时抖动和时间戳等重要信息，发送端应用根据这些信息可以估计出往返时延，并且可以根据数据报丢失概率和时延抖动情况动态调整发送速率，以改善网络拥塞状况，或者根据网络状况平滑地调整应用程序的服务质量。

RTSP实时流协议

作为一个应用层协议，RTSP提供了一个可供扩展的框架，它的意义在于使得实时流媒体数据的受控和点播变得可能。总的说来，RTSP是一个流媒体表示协议，主要用来控制具有实时特性的数据发送，但它本身并不传输数据，而是必须依赖于下层传输协议所提供的某些服务。RTSP可以对流媒体提供诸如播放、暂停、快进等操作，它负责定义具体的控制消息、操作方法、状态码等，此外还描述了与RTP间的交互操作。

RTSP在制定时较多地参考了HTTP/1.1协议，甚至许多描述与HTTP/1.1完全相同。RTSP之所以特意使用与HTTP/1.1类似的语法和操作，在很大程度上是为了兼容现有的Web基础结构，正因如此，HTTP/1.1的扩展机制大都可以直接引入到RTSP中。

由RTSP控制的媒体流集合可以用表示描述（Presentation Description）来定义，所谓表示是指流媒体服务器提供给客户机的一个或者多个媒体流的集合，而表示描述则包含了一个表示中各个媒体流的相关信息，如数据编码/解码算法、网络地址、媒体流的内容等。

虽然RTSP服务器同样也使用标识符来区别每一流连接会话（Session），但RTSP连接并没有被绑定到传输层连接（如TCP等），也就是说在整个RTSP连接期间，RTSP用户可打开或者关闭多个对RTSP服务器的可靠传输连接以发出RTSP 请求。此外，RTSP连接也可以基于面向无连接的传输协议（如UDP等）。

RTSP协议目前支持以下操作：

检索媒体 允许用户通过HTTP或者其它方法向媒体服务器提交一个表示描述。如表示是组播的，则表示描述就包含用于该媒体流的组播地址和端口号；如果表示是单播的， 为了安全在表示描述中应该只提供目的地

址。

邀请加入　　媒体服务器可以被邀请参加正在进行的会议，或者在表示中回放媒体，或者在表示中录制全部媒体或其子集，非常适合于分布式教学。

添加媒体　　通知用户新加入的可利用媒体流，这对现场讲座来讲显得尤其有用。与HTTP/1.1类似，RTSP请求也可以交由代理、通道或者缓存来进行处理。

3． JM86中的处理

涉及的函数：

流程图：

I帧和IDR帧的区别：

1． 在 H.264 中 I 帧并不具有随机访问的能力，这个功能由 IDR 承担。以前的标准中由 I 帧承担。

2． IDR 会导致 DPB （参考帧列表——这是关键所在）清空，而 I 不会。

3． I和IDR帧其实都是I帧,都是使用帧内预测的。但是IDR帧的作用是立刻刷新,使错误不致传播,从IDR帧开始,重新算一个新的序列开始编码。

4． IDR图像一定是I图像，但I图像不一定是IDR图像。一个序列中可以有很多的I图像，I图像之后的图像可以引用I图像之间的图像做运动参考。

原文地址：http://www.tichinese.com/Article/Video/200909/2142.html

- « 上一篇
- 下一篇 »

评论3

- 
  1楼：中山野鬼 发表于 2012-05-07 09:31 回复此评论
  呵呵。H264的东西，现在还有人在看啊?
- 2楼：duilib 发表于 2012-08-07 09:06 回复此评论

  
  引用来自"中山野鬼"的评论

  呵呵。H264的东西，现在还有人在看啊?

  大仙，应该看什么，H265吗?
-