H264 NALU分析

`NALU(Network Abstract Layer Unit)`

音视频编码在流媒体和网络领域占有重要地位；流媒体编解码流程大致如下图所示：

H264简介

H.264从1999年开始，到2003年形成草案，最后在2007年定稿有待核实。在ITU的标准里称为H.264，在MPEG的标准里是MPEG-4的一个组成部分–MPEG-4 Part 10，又叫Advanced Video Codec，因此常常称为MPEG-4 AVC或直接叫AVC。

H264 编解码解析

参考链接： H264 编解码协议详解;深入浅出理解视频编码H264结构;详细文档;H264编解码框图;

H264编码原理

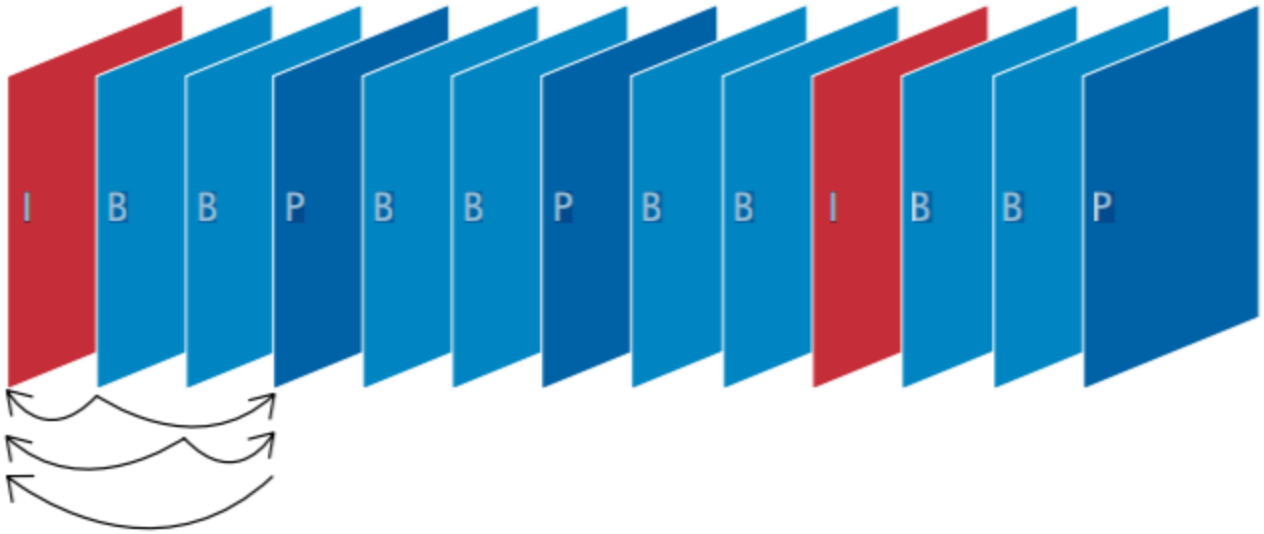
在音视频传输过程中，视频文件的传输是一个极大的问题；一段分辨率为1920*1080，每个像素点为RGB占用3个字节，帧率是25的视频，对于传输带宽的要求是：1920*1080*3*25/1024/1024=148.315MB/s，换成bps则意味着视频每秒带宽为1186.523Mbps，这样的速率对于网络存储是不可接受的。因此视频压缩和编码技术应运而生。

对于视频文件来说，视频由单张图片帧所组成，比如每秒25帧，但是图片帧的像素块之间存在相似性，因此视频帧图像可以进行图像压缩；H264采用了16*16的分块大小对，视频帧图像进行相似比较和压缩编码。如下图所示：

macroblock (16x16 pixels)

H264中的I帧、P帧和B帧

H26使用帧内压缩和帧间压缩的方式提高编码压缩率；H264采用了独特的I帧、P帧和B帧策略来实现，连续帧之间的压缩；



如上图所示；

帧的分类	中文	意义
I帧	帧内编码帧 intra picture	I 帧通常是每个 GOP（MPEG 所使用的一种视频压缩技术）的第一个帧，经过适度地压缩，做为随机访问的参考点，可以当成图象。I帧可以看成是一个图像经过压缩后的产物。 自身可以通过视频解压算法解压成一张单独的完整的图片。

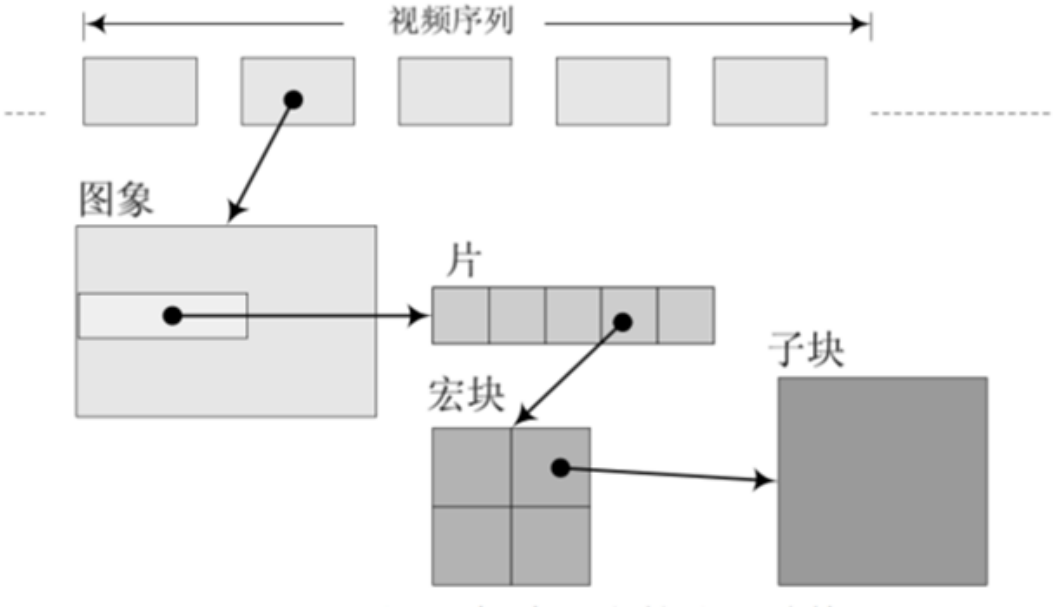
P帧	前向预测编码帧 predictive-frame	通过充分将低于图像序列中前面已编码帧的时间冗余信息来压缩传输数据量的编码图像，也叫预测帧。 需要参考其前面的一个I frame 或者P frame来生成一张完整的图片。
B帧	双向预测帧 bi-directional interpolated prediction frame	既考虑与源图像序列前面已编码帧，也顾及源图像序列后面已编码帧之间的时间冗余信息来压缩传输数据量的编码图像，也叫双向预测帧。 则要参考其前一个I或者P帧及其后面的一个P帧来生成一张完整的图片。

压缩率 B > P > I

H264编码结构解析

H264除了实现了对视频的压缩处理之外，为了方便网络传输，提供了对应的视频编码和分片策略；类似于网络数据封装成IP帧，在H264中将其称为组(GOP, group of pictures)、片（slice）、宏块（Macroblock）这些一起组成了H264的码流分层结构；H264将其组织成为序列(GOP)、图片(pictrue)、片(Slice)、宏块(Macroblock)、子块(subblock)五个层次。

GOP （图像组）主要用作形容一个IDR帧 到下一个IDR帧之间的间隔了多少个帧。



H264将视频分为连续的帧进行传输，在连续的帧之间使用I帧、P帧和B帧。同时对于帧内而言，将图像分块为片、宏块和字块进行分片传输；通过这个过程实现对视频文件的压缩包装。

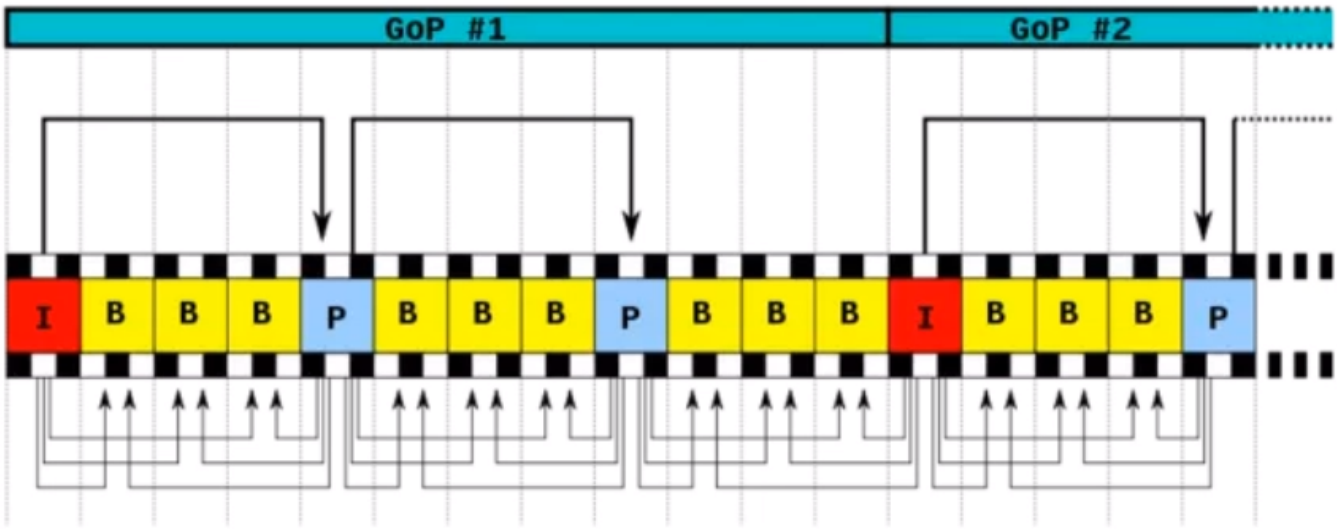
IDR（Instantaneous Decoding Refresh，即时解码刷新）

一个序列的第一个图像叫做 IDR 图像（立即刷新图像），IDR 图像都是 I 帧图像。
I和IDR帧都使用帧内预测。I帧不用参考任何帧，但是之后的P帧和B帧是有可能参考这个I帧之前的帧的。IDR就不允许这样。比如（解码的顺序）：

- IDR1 P4 B2 B3 P7 B5 B6 I10 B8 B9 P13 B11 B12 P16 B14 B15 这里的B8可以跨过I10去参考P7
  原始图像： IDR1 B2 B3 P4 B5 B6 P7 B8 B9 I10
- IDR1 P4 B2 B3 P7 B5 B6 IDR8 P11 B9 B10 P14 B11 B12 这里的B9就只能参照IDR8和P11，不可以参考IDR8前面的帧

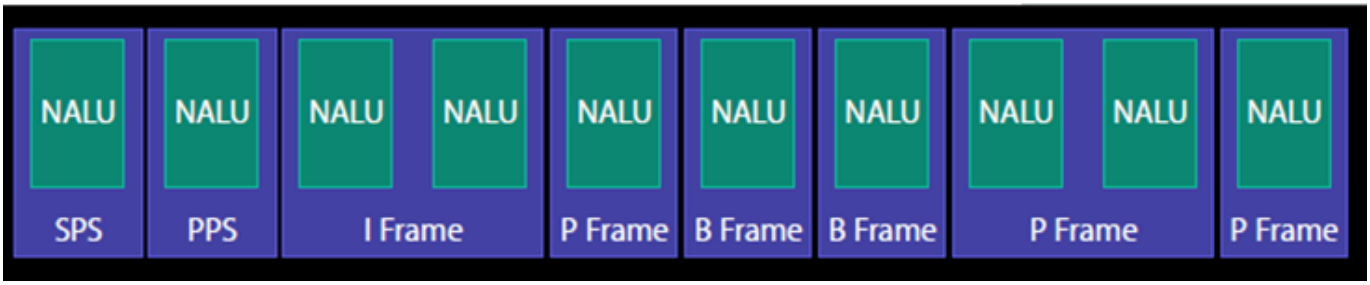
其核心作用是，是为了解码的重同步，当解码器解码到 IDR 图像时，立即将参考帧队列清空，将已解码的数据全部输出或抛弃，重新查找参数集，开始一个新的序列。这样，如果前一个序列出现重大错误，在这里可以获得重新同步的机会。IDR图像之后的图像永远不会使用IDR之前的图像的数据来解码。

下面是一个H264码流的举例（从码流的帧分析可以看出来B帧不能被当做参考帧）



I0 B40 B80 B120 P160
I0 B160

NALU



SPS：序列参数集，SPS中保存了一组编码视频序列(Coded video sequence)的全局参数。
PPS：图像参数集，对应的是一个序列中某一幅图像或者某几幅图像的参数。
I帧：帧内编码帧，可独立解码生成完整的图片。
P帧: 前向预测编码帧，需要参考其前面的一个I 或者B 来生成一张完整的图片。
B帧: 双向预测内插编码帧，则要参考其前一个I或者P帧及其后面的一个P帧来生成一张完整的图片。

发I帧之前，至少要发一次SPS和PPS。

NALU结构

H.264原始码流(裸流)是由一个接一个NALU组成，它的功能分为两层，VCL(视频编码层)和NAL(网络提取层)：

- VCL：包括核心压缩引擎和块，宏块和片的语法级别定义，设计目标是尽可能地独立于网络进行高效的编码；
- NAL：负责将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，覆盖了所有片级以上的语法级别

在VCL进行数据传输或存储之前，这些编码的VCL数据，被映射或封装进NAL单元。（NALU）

> **一个NALU = 一组对应于视频编码的NALU头部信息 + 一个原始字节序列负荷(RBSP,Raw Byte Sequence Payload).**

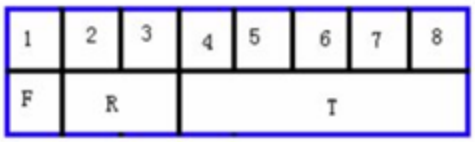
NALU结构单元的主体结构如下所示；一个原始的H.264 NALU单元**通常由[StartCode] [NALU Header] [NALU Payload]**三部分组成，其中 Start Code 用于标示这是一个NALU 单元的开始，必须是"00 00 00 01" 或"00 00 01"，除此之外基本相当于一个NAL header + RBSP;

（对于FFmpeg解复用后，MP4文件读取出来的packet是不带startcode，但TS文件读取出来的packet带了startcode）

解析NALU

每个NAL单元是一个一定语法元素的可变长字节字符串，包括包含一个字节的头信息（用来表示数据类型），以及若干整数字节的负荷数据。

NALU头信息（一个字节）：



其中：

- **T为负荷数据类型，占5bit**

  nal_unit_type：这个NALU单元的类型,1～12由H.264使用，24～31由H.264以外的应用使用

- **R为重要性指示位，占2个bit**

  nal_ref_idc.：取00~11,似乎指示这个NALU的重要性,如00的NALU解码器可以丢弃它而不影响图像的回放,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。

- **最后的F为禁止位，占1bit**

  forbidden_zero_bit： 在 H.264 规范中规定了这一位必须为 0.

H.264标准指出，当数据流是储存在介质上时，在每个NALU 前添加起始码：0x000001 或0x00000001，用来指示一个NALU 的起始和终止位置：

- 在这样的机制下，在码流中检测起始码，作为一个NALU得起始标识，当检测到下一个起始码时，当前NALU结束。
- 3字节的0x000001只有一种场合下使用，就是一个完整的帧被编为多个slice（片）的时候，包含这些slice的NALU 使用3字节起始码。其余场合都是4字节0x00000001的。

例子：

0x00 00 00 01 67 …

0x00 00 00 01 68 …

0x00 00 00 01 65 …

67：

二进制：0110 0111

00111 = 7（十进制）

nal_unit_type	NAL 单元和 RBSP 语法结构的内容	
0	未指定	
1	一个非IDR图像的编码条带 slice_layer_without_partitioning_rbsp( )	
2	编码条带数据分割块A slice_data_partition_a_layer_rbsp( )	
3	编码条带数据分割块B slice_data_partition_b_layer_rbsp( )	
4	编码条带数据分割块C slice_data_partition_c_layer_rbsp( )	
5	IDR图像的编码条带(片) slice_layer_without_partitioning_rbsp( )	
6	辅助增强信息 (SEI) sei_rbsp( )	
7	序列参数集 seq_parameter_set_rbsp( )	
8	图像参数集 pic_parameter_set_rbsp( )	
9	访问单元分隔符 access_unit_delimiter_rbsp( )	
10	序列结尾 end_of_seq_rbsp( )	
11	流结尾 end_of_stream_rbsp( )	
12	填充数据 filler_data_rbsp( )	
13	序列参数集扩展	

	seq_parameter_set_extension_rbsp( )	
14...18	保留	
19	未分割的辅助编码图像的编码条带 slice_layer_without_partitioning_rbsp ( )	
20...23	保留	
24...31	未指定	

对于NALU分析这节课主要关注5/6/7/8 四种类型。

H264 annexb模式

H264有两种封装

- 一种是annexb模式，传统模式，有startcode，SPS和PPS是在ES中
- 一种是mp4模式，一般mp4 mkv都是mp4模式，没有startcode，SPS和PPS以及其它信息被封装在container中，每一个frame前面4个字节是这个frame的长度

很多解码器只支持annexb这种模式，因此需要将mp4做转换：在ffmpeg中用h264_mp4toannexb_filter可以做转换

实现：

```
const AVBitStreamFilter *bsfilter = av_bsf_get_by_name("h264_mp4toannexb");
AVBSFContext *bsf_ctx = NULL;
// 2 初始化过滤器上下文
av_bsf_alloc(bsfilter, &bsf_ctx); //AVBSFContext;
// 3 添加解码器属性
avcodec_parameters_copy(bsf_ctx->par_in, ifmt_ctx->streams[videoindex]->codecpar);
av_bsf_init(bsf_ctx);
```

补充讲解

GOP group of pictures

GOP 指的就是两个I帧之间的间隔. 比较说GOP为120,如果是720 p60 的话,那就是2s一次I帧.
在视频编码序列中，主要有三种编码帧：I帧、P帧、B帧，如下所示：

1. I帧即Intra-coded picture（帧内编码图像帧），不参考其他图像帧，只利用本帧的信息进行编码。
2. P帧即Predictive-codedPicture（预测编码图像帧），利用之前的I帧或P帧，采用运动预测的方式进行帧间预测编码。
3. B帧即Bidirectionallypredicted picture（双向预测编码图像帧)，提供最高的压缩比，它既需要之前的图像帧(I帧或P帧)，也需要后来的图像帧(P帧)，采用运动预测的方式进行帧间双向预测编码。

在视频编码序列中，GOP即Group of picture（图像组），指两个I帧之间的距离，Reference（参考周期）指两个P帧之间的距离。一个I帧所占用的字节数大于一个P帧，一个P帧所占用的字节数大于一个B帧。

所以在码率不变的前提下，GOP值越大，P、B帧的数量会越多，平均每个I、P、B帧所占用的字节数就越多，也就更容易获取较好的图像质量；Reference越大，B帧的数量越多，同理也更容易获得较好的图像质量。

需要说明的是，通过提高GOP值来提高图像质量是有限度的，在遇到场景切换的情况时，H.264编码器会自动强制插入一个I帧，此时实际的GOP值被缩短了。另一方面，在一个GOP中，P、B帧是由I帧预测得到的，当I帧的图像质量比较差时，会影响到一个GOP中后续P、B帧的图像质量，直到下一个GOP开始才有可能得以恢复，所以GOP值也不宜设置过大。
同时，由于P、B帧的复杂度大于I帧，所以过多的P、B帧会影响编码效率，使编码效率降低。另外，过长的GOP还会影响Seek操作的响应速度，由于P、B帧是由前面的I或P帧预测得到的，所以Seek操作需要直接定位，解码某一个P或B帧时，需要先解码得到本GOP内的I帧及之前的N个预测帧才可以，GOP值越长，需要解码的预测帧就越多，seek响应的时间也越长。

H.264中的I帧，B帧和P帧

在H264中的图像以序列为单位进行组织，一个序列是一段图像编码后的数据流，以I帧开始，到下一个I帧结束。
IDR图像：一个序列的第一个图像叫做IDR图像（立即刷新图像），IDR 图像都是I帧图像。
H.264引入IDR图像是为了解码的重同步，当解码器解码到IDR图像时，立即将参考帧队列清空，将已解码的数据全部输出或抛弃，重新查找参数集，开始一个新的序列。这样，如果前一

个序列出现重大错误，在这里获得重新同步的机会。IDR图像之后的图像永远不会使用IDR之前的图像数据来解码。
一个序列就是一段内容差别不是很大的图像编码后生成的一串数据流。当运动变化比较少的时候，一个序列可以很长，因为运动变化的少就代表图像画面的内容变动很小，所以就可以编一个I帧，然后一直P帧、B帧了。当运动变化多时，可能一个序列就比较短了，比如就包含一个I帧和3、4个P帧。

I P B三种帧的说明

1、I帧

I帧:帧内编码帧 ，I帧表示关键帧，你可以理解为这一帧画面的完整保留；解码时只需要本帧数据就可以完成（因为包含完整画面）
I帧特点:

- 1) 它是一个全帧压缩编码帧。它将全帧图像信息进行JPEG压缩编码及传输;
- 2) 解码时仅用I帧的数据就可重构完整图像;
- 3) I帧描述了图像背景和运动主体的详情;
- 4) I帧不需要参考其他画面而生成;
- 5) I帧是P帧和B帧的参考帧(其质量直接影响到同组中以后各帧的质量);
- 6) I帧是帧组GOP的基础帧(如果为IDR则为第一帧),在一组中只有一个IDR帧，一个或多个I帧(包括IDR帧);
- 7) I帧不需要考虑运动矢量;
- 8) I帧所占数据的信息量比较大。

2、P帧

P帧:前向预测编码帧。P帧表示的是这一帧跟之前的一个关键帧（或P帧）的差别，解码时需要用之前缓存的画面叠加上本帧定义的差别，生成最终画面。（也就是差别帧，P帧没有完整画面数据，只有与前一帧的画面差别的数据）
P帧的预测与重构:P帧是以I帧为参考帧,在I帧中找出P帧“某点”的预测值和运动矢量,取预测差值和运动矢量一起传送。在接收端根据运动矢量从I帧中找出P帧“某点”的预测值并与差值相加以得到P帧“某点”样值,从而可得到完整的P帧。
P帧特点:

- 1) P帧是I帧后面相隔1~2帧的编码帧;
- 2) P帧采用运动补偿的方法传送它与前面的I或P帧的差值及运动矢量(预测误差);
- 3) 解码时必须将I帧中的预测值与预测误差求和后才能重构完整的P帧图像;
- 4) P帧属于前向预测的帧间编码。它只参考前面最靠近它的I帧或P帧;
- 5) P帧可以是其后面P帧的参考帧,也可以是其前后的B帧的参考帧;
- 6) **由于P帧是参考帧**,它可能造成解码错误的扩散;
- 7) 由于是差值传送,P帧的压缩比较高。

3、B帧

B帧:双向预测内插编码帧。B帧是双向差别帧，也就是B帧记录的是本帧与前后帧的差别（具体比较复杂，有4种情况，但我这样说简单些），换言之，要解码B帧，不仅要取得之前的缓存画面，还要解码之后的画面，通过前后画面的与本帧数据的叠加取得最终的画面。B帧压缩率高，但是解码时CPU会比较累。

B帧的预测与重构

B帧以前面的I或P帧和后面的P帧为参考帧,“找出”B帧“某点”的预测值和两个运动矢量,并取预测差值和运动矢量传送。接收端根据运动矢量在两个参考帧中“找出(算出)”预测值并与差值求和,得到B帧“某点”样值,从而可得到完整的B帧。

B帧特点

- 1）B帧是由前面的I或P帧和后面的P帧来进行预测的;
- 2）B帧传送的是它与前面的I或P帧和后面的P帧之间的预测误差及运动矢量;
- 3）B帧是双向预测编码帧;
- 4）B帧压缩比最高,因为它只反映两参考帧间运动主体的变化情况,预测比较准确;
- 5）**B帧不是参考帧**,不会造成解码错误的扩散。

注:I、B、P各帧是根据压缩算法的需要，是人为定义的,它们都是实实在在的物理帧。一般来说，I帧的压缩率是7（跟JPG差不多），P帧是20，B帧可以达到50。可见使用B帧能节省大量空间，节省出来的空间可以用来保存多一些I帧，这样在相同码率下，可以提供更好的画质。