

**LAS: Live Adaptive Streaming**

LAS1.0

2020 年 6 月 21 日周超 快手

郭亮 快手

于冰 快手

目录

1. 前言 1
2. 架构 2
3. 媒体呈现描述 3
4. LAS 请求描述 6
5. LAS 服务描述 8
6. LAS 客户端描述 13

7. 附录… 19

1. 前言

LAS (Live Adaptive Streaming) 是基于流式的直播多码率自适应标准，描述了基于流式的直播多码率自适应框架与实现规范。

参考代码：https://github.com/KwaiVideoTeam/las 官方网站：https://las-tech.org.cn

1.1.优势

LAS 具有以下优势：

* 1. 低延迟：LAS 实现基于流式的直播多码率自适应，实现帧级传输，降低端到端延迟
  2. 易扩展：LAS 直播多码率自适应框架，与传统的流式非多码率框架（例如 HTTP-FLV）完全兼容；LAS 支持多种协议，例如 HTTP、 QUIC、

WebRTC 等

* 1. 易部署：各大云厂商均支持，经过大规模验证，有大量优秀案例，例如快手等
  2. 高效性：请求数量少，降低开销。LAS 只有在起播或发生码率切换时才需要发送请求
  3. .核心内容

LAS 标准的核心内容包括

* + 1. 媒体呈现描述：描述了基于流式的直播多码率自适应标准的基本语义元素
    2. LAS 请求描述：描述了基于流式的直播多码率自适应标准中，不同场景下的请求生成方式
    3. LAS 服务描述：描述了服务端/云端支持基于流式的直播多码率自适应标准的处理逻辑
    4. LAS 客户端描述：LAS 客户端的具体实现，不作为 LAS 标准的范畴。

LAS 仅给出推荐实现架构与自适应算法策略

1. 架构

本部分定义了 LAS 的架构图，见图 2.1。LAS 标准主要规定了基于流式的直播多码率自适应的媒体呈现描述、LAS 请求描述、LAS 服务描述与推荐的

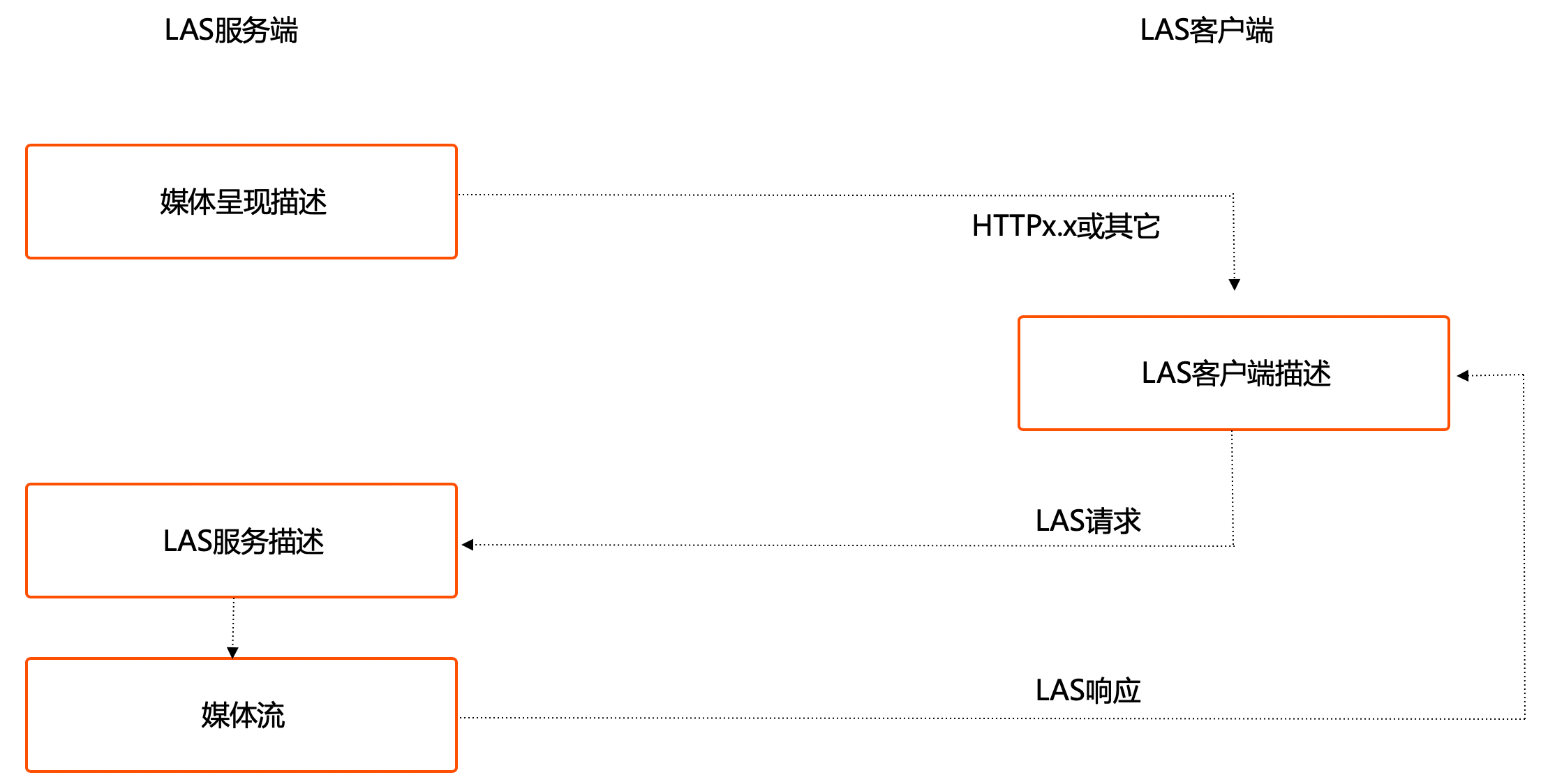
LAS 客户端架构及直播多码率无缝自适应逻辑。

图 2.1 LAS 架构

2.1.媒体呈现描述

媒体呈现描述是为了提供流媒体业务而对媒体呈现（Media Presentation， 简称 MP）进行的格式化描述。具体而言，本部分定义了媒体流的资源标识符格式及被标识的资源在媒体呈现中的上下文。

* 1. LAS 请求描述

LAS 请求是 LAS 客户端向服务端请求媒体流的请求规范。具体而言，本部分详细定义了请求的生成规范和请求规范。

* 1. LAS 服务描述

LAS 服务描述规定了服务端对媒体流转码、缓存等特性的规范，以及对 LAS

请求逻辑的响应规范、异常处理规范。

* 1. LAS 客户端描述

LAS 客户端是 LAS 请求发起与媒体处理的载体，具体实现不属于 LAS 标准的范畴。LAS 标准仅给出推荐的客户端架构、直播多码率自适应策略、无缝切换方案等。

1. 媒体呈现描述

媒体呈现（MP）是由客户端访问、用于向用户提供流媒体服务的一组数据的

集合。其中包括了已编码并可传输的媒体流及其恰当描述。媒体呈现描述

（MPD）是一个包含元数据的 JSON 文档。LAS 客户端通过这些元数据来构建获取媒体流的 LAS 请求，并向用户提供流媒体服务。MPD 示例见附录

A。

* 1. .层次化数据模型

采用 JSON 格式，定义了媒体呈现描述的元素与语义。

* + 1. MPD 的语义

MPD 元素的语义见表 3.1

表 3.1 MPD 语义表

|  |  |  |
| --- | --- | --- |
| 元素或属性名称 | 用法 | 描述 |
| MPD |  | 媒体呈现描述的根元素 |
| @version | 必选 | 版本号 |
| @adaptationSet | 必须，包含至少一个  @representation | 自适应集合  每个 MPD 由一个或多个  自适应集合组成 |

* + 1. adaptationSet 的语义

adaptationSet 元素的语义见表 3.2

表 3.2 adaptationSet 元素的语义

|  |  |  |
| --- | --- | --- |
| 元素或属性名称 | 用法 | 描述 |
| @duration | 必选 | 媒体流 GOP 的长度，单位为毫秒 |
| @id | 必选 | adaptationSet 的唯一标识号 |
| @representation | 必选 | 媒体表示的集合，包含一个或多个媒体表示 |

表 3.3 媒体表示元素的语义

|  |  |  |
| --- | --- | --- |
| 元素或属性名称 | 用法 | 描述 |
| @id | 必选 | 每个媒体表示的唯一标识号 |
| @codec | 必选 | 音视频流的编码方式 |
| @url | 必选 | 媒体表示的 URL 地址 |
| @backupUrl | 必选 | 媒体表示的备用 URL 地址 |
| @host | 可选 | 媒体表示的域名 |
| @maxBitrate | 必选 | 媒体表示的编码码率 |
| @avgBitrate | 可选 | 媒体表示的平均码率 |
| @width | 可选 | 媒体表示的宽度 |
| @height | 可选 | 媒体表示的高度 |
| @frameRate | 可选 | 媒体表示的帧率 |
| @qualityType | 可选 | 媒体表示的质量类型 |
| @qualityTypeName | 可选 | 媒体表示的质量类型展示字段 |
| @hidden | 可选 | 媒体表示隐藏选项  设定为 true，则对应的媒体表示不外显，用户无法选择，只能通过自适应功能选中；  设定为 false，则对应的媒体表示外显，用户可手动选择 |
| @disabledFromAdaptive | 可选 | 媒体表示自适应选项  设定为 false，则对应的媒体表示对于自适应功能可见，能被自适应功能选中； |

|  |  |  |
| --- | --- | --- |
|  |  | 设定为 true，则对应的媒体表示对于自适应功能不可见，不能被自适应功能选中； |
| @defaultSelected | 可选 | 默认档功能选项  @defaultSelected 为true 的媒体表示启播默认播放  注：   1. 所有@representation 中，最多只能出现一个 媒 体 表 示 的   @defaultSelected 为  true   1. 没有@representation 设置 defaultSelected 为 true 时，自动选择媒体表示启播播放 |

1. LAS 请求描述

LAS 请求描述了客户端向服务端发送请求的具体格式与含义，LAS 请求用于向媒体服务器请求媒体流。

LAS 基本请求格式定义为媒体流 URL 地址加扩展字段的形式， 即：

url&extParam。详细的 LAS 请求示例见附录B。

表 4.1 LAS 请求元素的语义

|  |  |  |
| --- | --- | --- |
| 属性 | 用法 | 描述 |
| @url | 必选 | 媒体表示的地址  指向媒体表示的地址，从媒体呈现描述文件中获取 |
| @extParam | 可选 | 扩展参数。  指定不同的请求方式，从  而实现不同的功能 |

表 4.2 exeParam 元素的语义

|  |  |  |
| --- | --- | --- |
| 属性 | 用法 | 描述 |
| @audioOnly | 可选，默认值 false | 音频参数  当设定为 true 时，只拉取纯音频流，否则，拉取音频流和视频流 |
| @startPts | 可选，默认值服务器可 | 拉流位置参数 |
|  | 配置。int64\_t 类型 | 当设定@startPts=0：非 |
|  |  | 纯音频模式时，从最新的 |
|  |  | 视频 I 帧开始拉流；纯音 |
|  |  | 频模式时，从最新的音频 |
|  |  | 帧开始拉流 |
|  |  | 当设定@startPts>0：从 |
|  |  | pts 等于@startPts 的媒 |
|  |  | 体帧开始拉流 |
|  |  | 当设定@startPts<0：拉 |
|  |  | 取 缓 存 长 度 为 |

|  |  |  |
| --- | --- | --- |
|  |  | |@startPts| 毫秒的媒体数据 |

1. LAS 服务描述

LAS 服务描述规定了服务端（CDN 或自建媒体服务器）对媒体流转码、缓存等特性的规范，以及对 LAS 请求逻辑的响应规范、异常处理规范。

注：在本规范中，服务端和 CDN 作为可无差交换使用的替换说法。目前国内主流云厂商均支持该规范。

* 1. .转码规范

转码规范要求转码服务不得对原始流中 I 帧的 pts 做任何修改。具体而言， 输入任意一路原始视频，经过转码服务后，输出多路不同分辨率、码率的视频流，各视频的 I 帧严格对齐，如下图所示：

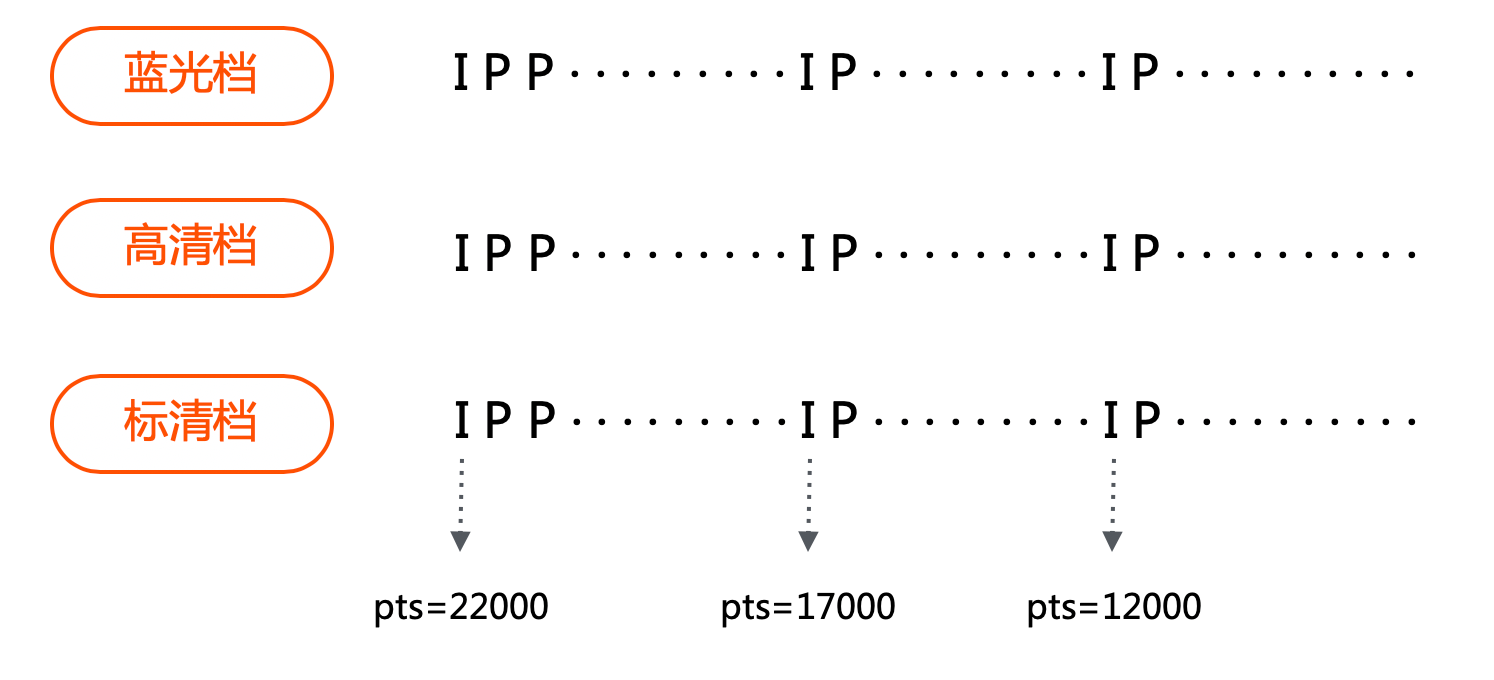


图 5.1 转码规范示意图

* 1. .缓存规范
     1. 缓存时长配置

描述了在服务端最少需要缓存的数据长度，单位为毫秒。

表 5.1 缓存时长元素的含义

|  |  |  |
| --- | --- | --- |
| 属性 | 用法 | 描述 |
| @maxCachedDuration | 可选，默认值由  服务端配置 | 指定服务端最多缓存的数据长度，单位 ms。 |

|  |  |  |
| --- | --- | --- |
|  |  | 若@ maxCachedDuration >0， 则缓存@ maxCachedDuration 毫秒的媒体数据，否则缓存时长为默认值的媒体数据 |

* + 1. 缓存时长计算

支持依据音视频流分开计算缓存时长。

默认使用视频流计算缓存时长，没有视频流时使用音频流计算缓存时长。

* + 1. 时间戳回退处理
       1. 缓存时间戳回退定义

以下任意一种情况均称为发生缓存时间戳回退

I． 当缓存中有视频时，则以 I 帧 pts 为计算点(只考虑 I 帧的 pts)，缓存

中I 帧pts 序列有非单调递增的情况

II． 当缓存中没有视频时，则以所有音频 pts 为计算点(只考虑音频帧的

pts)，缓存中音频帧的 pts 序列有非单调递增的情况

* + - 1. 时间戳回退处理

定义一：latestVideoPts，指缓存中最新的视频帧的 pts

定义二：latestAudioPts，指缓存中最新的音频帧的 pts

定义三：有效缓存区，包括如下两种情况：a) 5.2.3.1 中情况 I 的处理：当缓存中有视频时，以视频 I 帧pts 作为计算点，从最后一个单调递增点的 I 帧到

latestVideoPts 的所有媒体帧作为有效缓存区; b) 5.2.3.1 中情况 II 的处理： 当缓存中没有视频时，以音频 pts 作为计算点，从最后一个单调递增的音频帧到 latestAudioPts 的所有音频帧作为有效缓存区。

* 1. LAS 请求逻辑处理规范

LAS 请求逻辑处理规范描述了服务端收到 LAS 请求后，具体的响应逻辑。

* + 1. LAS 请求中， 可选项@audioOnly 缺省或@audioOnly=false ，同时

@startPts 缺省

使用@startPts 的默认值（@startPts= defaultStartPts），向 LAS 客户端发送LAS 请求中@url 指定的视频流和音频流，规则如下：

I． 当缓存中有视频时：从当前有效缓存区中 pts 最接近 latestVideoPts

- |defaultStartPts|的 I 帧开始发送媒体流

II． 当缓存 中 没 有视频 时： 从当 前 有效缓存区 中 pts 最接近

latestAudioPts - |defaultStartPts| 的音频帧开始发送媒体流

* + 1. LAS 请求中，可选项@audioOnly=true，同时@startPts 缺省

使用@startPts 的默认值（@startPts= defaultStartPts），向 LAS 客户端发送LAS 请求中@url 指定的音频流，规则如下：

* + - 1. 从当前有效缓存区中 pts 最接近 latestAudioPts - |defaultStartPts|

的音频帧开始发送媒体流

* + 1. LAS 请求中， 可选项@audioOnly 缺省或@audioOnly=false ，同时

@startPts=0

使用@startPts=0，向 LAS 客户端发送 LAS 请求中@url 指定的视频流和音频流，规则如下：

* + - 1. 当缓存中有视频时：从当前有效缓存区中 pts 最接近 latestVideoPts

的I 帧开始发送媒体流

* + - 1. 当缓存中没有视频时：从当前有效缓存区中pts 最接近latestAudioPts

的音频帧开始发送媒体流

* + 1. LAS 请求中，可选项@audioOnly=true，同时@startPts=0

使用@startPts=0，向 LAS 客户端发送 LAS 请求中@url 指定的音频流，规则如下：

* + - 1. 从当前有效缓存区中 pts 最接近 latestAudioPts 的音频帧开始发送媒体流
    1. LAS 请求中， 可选项@audioOnly 缺失或@audioOnly=false ，同时

@startPts<0

使用@startPts<0，向 LAS 客户端发送 LAS 请求中@url 指定的视频流和音频流，规则如下：

* + - 1. 当缓存中有视频时：从当前有效缓存区中 pts 最接近 latestVideoPts

- |@startPts|的 I 帧开始发送媒体流

* + - 1. 当缓存中没有视频 时： 从当前 有效缓 存区中 pts 最接近

latestAudioPts - |@startPts|的音频帧开始发送媒体流

* + 1. LAS 请求中，可选项@audioOnly=true，同时@startPts<0

使用@startPts<0，向 LAS 客户端发送 LAS 请求中@url 指定的音频流，规则如下：

* + - 1. 从当前有效缓存区中 pts 最接近 latestAudioPts - |@startPts|的音频

帧开始发送媒体流

* + 1. LAS 请求中， 可选项@audioOnly 缺省或@audioOnly=false ，同时

@startPts>0

使用@startPts>0，向 LAS 客户端发送 LAS 请求中@url 指定的视频流和音频流，规则如下：

* + - 1. 当缓存中有视频且没有时间戳回退时: 从 pts 最小的 I 帧开始，沿着

pts 增大的方向，找到 pts 最大且满足 pts <= @startPts 的 I 帧开始发送媒体流；若找不到满足 pts <= @startPts 的I 帧，则找到第一个

pts 满足 pts > @startPts 的I 帧开始发送媒体流；若均找不到，则等

待第一个 pts 大于等于@startPts 的I 帧到来再开始发送媒体流

* + - 1. 当缓存中有视频且有时间戳回退时，则从最新的 I 帧开始发送媒体流
      2. 当缓存中没有视频且没有时间戳回退时: 从 pts 最小的音频帧开始， 沿着 pts 增大的方向，找到第一个 pts >= @startPts 的音频帧开始发送媒体流，若找不到满足 pts >= @startPts 的音频帧，则等待第一个

pts 大于等于@startPts 的音频帧或 I 帧到来再开始发送媒体流

* + - 1. 当缓存中没有视频且有时间戳回退时，则从最新的音频帧开始发送媒体流
    1. LAS 请求中，可选项@audioOnly=true，同时@startPts>0

使用@startPts>0，向 LAS 客户端发送 LAS 请求中@url 指定的音频流，规则如下：

* + - 1. 当没有时间戳回退时: 从 pts 最小的音频帧开始，沿着 pts 增大的方向，找到第一个 pts >= @startPts 的音频帧开始发送媒体流，若找不到满足 pts >= @startPts 的音频帧，则等待第一个 pts 大于等于

@startPts 的音频帧到来再开始发送媒体流

* + - 1. 当有时间戳回退时，则从最新的音频帧开始发送媒体流
    1. 服务端内部回源

当边缘节点没有 LAS 请求所指定媒体流时，需要向上级节点回源拉取媒体流:

* + - 1. 第三方源站回源，必须携带@startPts 字段
      2. 内部节点回源，如果携带@startPts 字段时则按 LAS 请求，依据

@startPts 的 实 际 值 进 行 回 源 ， 否 则 依 据 默 认 值

@startPts=defaultStartPts

* + 1. 异常处理

当@startPts>0 时，如果有效缓存区不存在 pts 小于等于@startPts 的媒体帧，有两种处理方式：

I. 等待模式：如 5.3.7 和 5.3.8

II. 错误处理模式：当@startPts 超过有效缓存区中最大的 pts 一定阈值

（超时阈值 timeoutPts）时，判定为 LAS 请求错误。阈值 timeoutPts

支持配置

1. LAS 客户端描述

LAS 客户端的具体实现，不作为 LAS 标准的范畴。LAS 仅给出推荐实现架构与自适应算法策略。

* 1. LAS 客户端架构

如图所示，客户端的主要逻辑包括：

* + 1. MPD 解析：负责解析 MPD，获取相应的媒体信息，例如各媒体表示

的 URL、码率、id 等信息

* + 1. 下载器：负责媒体流的下载，并将媒体数据传递给解码渲染模块；同时收集网络状态信息，并传递给自适应策略模块
    2. 解码渲染：从下载器接收媒体数据，并解码渲染；同时将播放相关状态，例如缓冲区大、卡顿情况、丢帧情况等信息，传递给自适应策略模块
    3. 自适应策略：结合 MPD 解析获取的备选媒体表示集、下载器传递的网络状态、解码渲染模块反馈的播放状态等信息，综合决策最佳的媒体表示，并传递给下载器，进行媒体表示的下载切换。

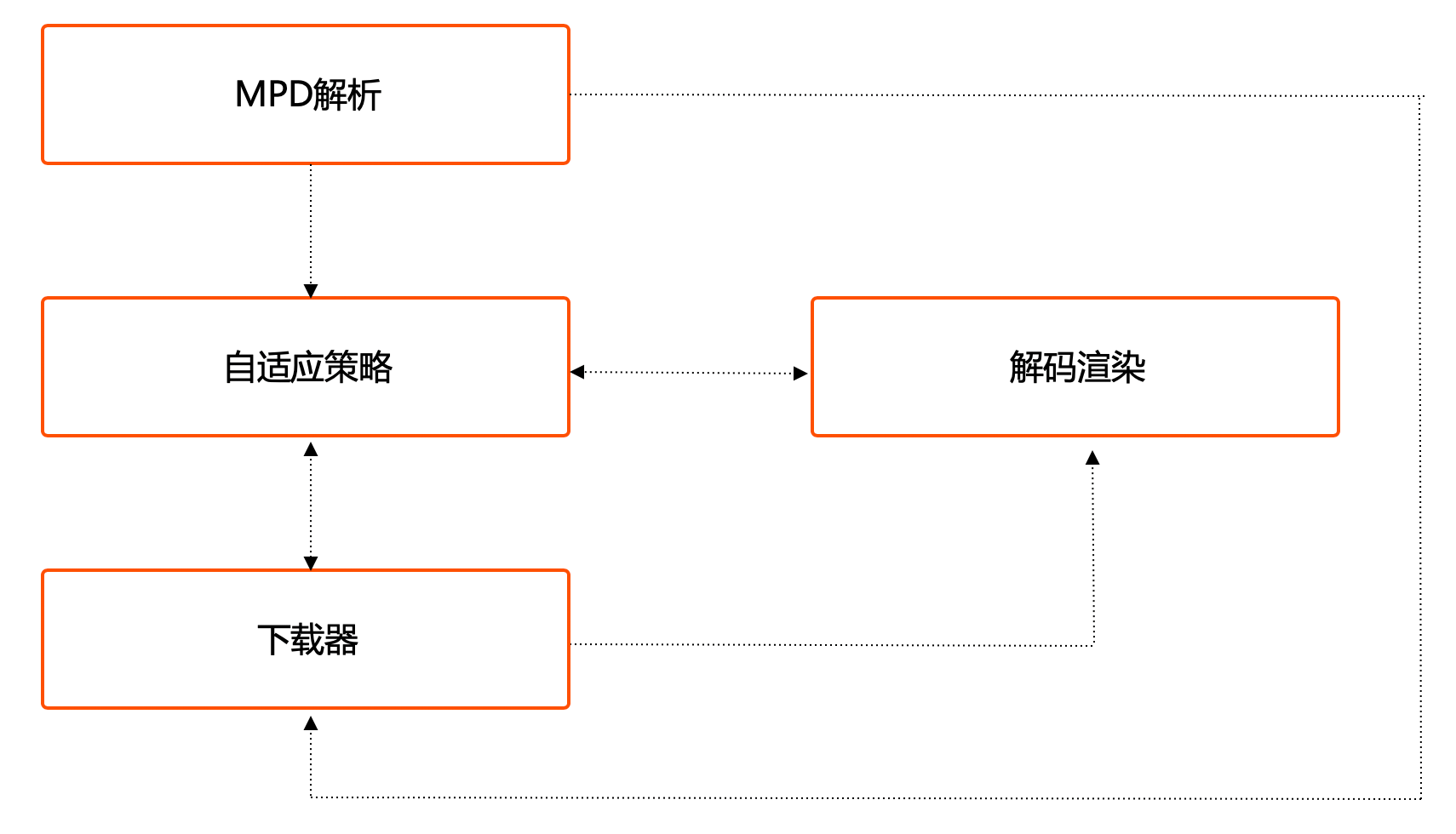


图 6.1 LAS 客户端逻辑示意图

* 1. MPD 解析

主要负责解析媒体呈现描述文件，并将各元素及属性传递给相应的模块。

* 1. .下载器

基于从自适应策略模块获取的媒体表示的相关信息，生成 LAS 请求，并发送到服务端。在 LAS 中，媒体表示采用流式传输，只有发生媒体表示切换时才需要重新发送请求。

**LAS: Live Adaptive Streaming**

LAS1.0

June 21 , 2020 Weekly Super Fast Hand

Guo Liang Quick

Yu Bing Kuaishou

directory

1. Preface.................................................................................................................... 1

2. Architecture.................................................................................................................... 2

3. Media presentation descriptions.................................................................................................... 3

4. LAS Request Description..................................................................................................... 6

5. LAS Service Description..................................................................................................... 8

6. LAS Client Description............................................................................................... 13

7. Appendix.... ............................................................................................................... 19

1. Preface

LAS (Live Adaptive Streaming) is a streaming-based live multi-bitrate adaptive standard that describes the multi-bitrate adaptive framework and implementation specifications for live streaming-based live broadcasts.

Reference code: https://github.com/KwaiVideoTeam/las Official website: https://las-tech.org.cn

1.1. Advantages

LAS offers the following advantages:

I. Low latency: LAS implements multi-bitrate adaptation based on streaming live broadcasting, realizes frame-level transmission, and reduces end-to-end latency

II. Easy to scale: LAS live multi-bitrate adaptive framework, and traditional streaming non-multi-bitrate frameworks (such as HTTP-FLV). Fully compatible; LAS supports multiple protocols such as HTTP, QUIC, etc

WebRTC, etc

III. Easy to deploy: All major cloud vendors support it, have been verified on a large scale, and have a large number of excellent cases, such as Kuaishou

IV. Efficiency: Fewer requests and lower overhead. LAS only needs to send requests when it starts or when a bitrate switch occurs

1.2 . Core content

The core content of the LAS standard includes

I. Media Rendering Description: Describes the basic semantic elements of the streaming live multi-bitrate adaptive standard

II. LAS Request Description: Describes the request generation method in different scenarios in the multi-bitrate adaptive standard for live streaming based on streaming

III. LAS Service Description: Describes the/ processing logic of the multi-bitrate adaptive standard for live streaming on the server/cloud to support streaming

IV. LAS Client Description: The specific implementation of the LAS client, not as LAS ry.

LAS only gives recommended implementation architectures and adaptive algorithm strategies

2. Architecture

This section defines the architecture diagram for laS, as shown in Figure 2.1. The LAS standard mainly stipulates the description of multi-bitrate adaptive media presentation, LAS request description, LAS service description and recommendation based on streaming

|  |
| --- |
|  |
|  |  |

LAS client architecture and live multi-bitrate seamless adaptive logic.

Figure 2.1 LAS Architecture

2.1. Media Presentation Description

Media Presentation Description Is a formatted description of Media Presentation ( MP) for the purpose of providing streaming media services. Specifically, this section defines the format of the resource identifier for the media stream and the context in which the identified resource is in media rendering.

2.2. LAS Request Description

A LAS request is a request specification for a LAS client to request a media stream from the server. Specifically, this section defines in detail the generation specification and request specification for the request.

2.3. LAS Service Description

The LAS service description specifies the specification of the server's features such as media streaming codes, caching, and laS

Response specifications and exception handling specifications for request logic.

2.4. LAS Client Description

The LAS client is the carrier of LAS request initiation and media processing, and the specific implementation does not belong to the scope of the LAS standard. /b15>The LAS standard only gives recommended client architecture, live multi-bitrate adaptive strategy, seamless switching scheme, etc.

3. Media presentation description

Media rendering (MP) is a set of data that is accessed by a client to provide streaming services to users

gather. This includes the media streams that have been encoded and can be transmitted and their appropriate descriptions. Media renders descriptions

(MPD) is a JSON document that contains metadata. LaS clients use this metadata to build LAS requests that get media streams and provide streaming services to users. An example of MPD is in the appendix

A。

3.1 . Hierarchical data model

In JSON format, it defines the elements and semantics of the media presentation description.

3.1.1. Semantics of MPD

The semantics of the MPD element are shown in Table 3.1

Table 3.1 MPD Semantic Tables

|  |  |  |
| --- | --- | --- |
| The name of the element or attribute | usage | description |
| MPD |  | The root element of the media rendering description |
| @version | Required | The version number |
| @adaptationSet | Must, contain at least one  @representation | Adaptive collections  Each MPD consists of one or more  Composed of adaptive collections |

3.1.2. Semantics of adaptationSet

The semantics of the adaptationSet element are shown in Table 3.2

Table 3.2 The semantics of the adaptationSet element

|  |  |  |
| --- | --- | --- |
| The name of the element or attribute | usage | description |
| @duration | Required | The length of the media stream GOP, in milliseconds |
| @id | Required | The unique identification number of the adaptationSet |
| @representation | Required | A collection of media representations that contains one or more media representations |

Table 3.3 The semantics of media representation elements

|  |  |  |
| --- | --- | --- |
| The name of the element or attribute | usage | description |
| @id | Required | A unique identification number represented by each medium |
| @codec | Required | The encoding of the audio and video stream |
| @url | Required | The URL address represented by the media |
| @backupUrl | Required | The alternate URL address represented by the media |
| @host | Optional | The domain name represented by the media |
| @maxBitrate | Required | The encoding bitrate represented by the media |
| @avgBitrate | Optional | The average bitrate represented by the media |
| @width | Optional | The width of the media representation |
| @height | Optional | The height of the media representation |
| @frameRate | Optional | The frame rate represented by the media |
| @qualityType | Optional | The type of quality that the media represents |
| @qualityTypeName | Optional | The Quality Type Presentation field for media representation |
| @hidden | Optional | Media represents a hidden option  Set to true, the corresponding media representation is not explicit, the user can not select, can only be selected through the adaptive function;  Set to false, the corresponding media represents the appearance, the user can manually select |
| @disabledFromAdaptive | Optional | Media represents adaptive options  Set to false, the corresponding media indicates that the adaptive function is visible and can be selected by the adaptive function; |

|  |  |  |
| --- | --- | --- |
|  |  | Set to true, the corresponding media indicates that it is not visible for the adaptive function and cannot be selected by the adaptive function; |
| @defaultSelected | Optional | Default file function options  @defaultSelected media for true indicates that playback is initiated by default    concentrate:  1. Of all @representation, only one media representation can appear  @defaultSelected is  true  2. When no @representation is set to true , the media is automatically selected to start playback |

4. LAS Request Description

A LAS request describes the specific format and meaning of a request that a client sends to the server, and a LAS request is used to request a media stream from a media server.

The LAS basic request format is defined as the form of a media streaming URL address plus an extended field, namely:

url&extParam。 Detailed examples of LAS requests are shown in Appendix B.

Table 4.1 Semantics of LAS Request Elements

|  |  |  |
| --- | --- | --- |
| attribute | usage | description |
| @url | Required | The address of the media representation  The address that points to the media representation, obtained from the media rendering profile |
| @extParam | Optional | Extend the parameters.  Specify different requests from  And implement different functions |

Table 4.2 Semantics of exeParam elements

|  |  |  |
| --- | --- | --- |
| attribute | usage | description |
| @audioOnly | Optional, the default value is false | Audio parameters  When set to true, only audio-only streams are pulled, otherwise, audio and video streams are pulled |
| @startPts | Optional, the default value server is available | Pull flow position parameter |
|  | disposition. int64\_t type | When set @startPts=0: Non |
|  |  | Audio-only mode when up-to-date |
|  |  | Video I frames begin to pull streams; pure tone |
|  |  | Frequency mode when from the latest audio |
|  |  | The frame starts pulling |
|  |  | When set @startPts> 0: From |
|  |  | pts equals @startPts medium |
|  |  | Body frames begin to pull streams |
|  |  | When set @startPts< 0: Pull |
|  |  | Take the slow memory length of as |

|  |  |  |
| --- | --- | --- |
|  |  | |@startPts| Milliseconds of media data |

5. LAS Service Description

The LAS service description specifies the specifications for the media streaming code, caching, and other features on the server side (CDN or self-built media server), as well as the response specifications and exception handling specifications for the LAS request logic.

Note: In this specification, the server and CDN are used as alternatives to differenceless swapping. At present, mainstream cloud vendors in China support this specification.

5.1 . Transcoding specifications

The transcoding specification requires that the transcoding service must not make any modifications to the pts of the I frames in the original stream. Specifically, after entering any source video, after transcoding service, multiple video streams of different resolutions and bitrates are output, and the I frames of each video are strictly aligned, as shown in the following figure:

Figure 5.1 Transcoding specification schematic

5.2 . Cache specification

5.2.1. Cache Duration Configuration

Describes the minimum length of data that needs to be cached on the server side, in milliseconds.

Table 5.1 The Meaning of the Cache Duration Element

|  |  |  |
| --- | --- | --- |
| attribute | usage | description |
| @maxCachedDuration | Optional, the default value is by  Server-side configuration | Specifies the maximum cached data length on the server side, in ms. |

|  |  |  |
| --- | --- | --- |
|  |  | If @maxCachedDuration >0, the media data for @maxCachedDuration milliseconds is cached, otherwise the media data with the default duration is cached |

5.2.2. Cache Duration Calculation

Supports calculating the cache duration separately based on the audio and video streams.

By default, the cache length is calculated using a video stream, and the cache length is calculated when there is no video stream.

5.2.3. Timestamp Fallback Processing

5.2.3.1. Cache Timestamp Fallback Definitions

Either of the following scenarios is referred to as cache timestamp fallback

I． When there is video in the cache, the I-frame pts is used as the calculation point (only the pts of the I-frame are considered) and the cache is taken

There is a non-monotonic increment in the I-frame pts sequence

II． When there is no video in the cache, all audio pts are used as the calculation point (only the audio frame is considered.)

pts) , the pts sequence of audio frames in the cache has a non-monotonic increment

5.2.3.2. Timestamp Fallback Processing

Definition 1: latestVideoPts refers to the pts of the latest video frames in the cache

Definition two: latestAudioPts, which refers to the pts of the latest audio frames in the cache

Definition three: A valid buffer, including two cases: a) The handling of case I in 5.2.3.1: when there is video in persist, the video frames pts As a calculation point, the I frame from the last monotonic increment point to

latestVideoPts all media frames as valid buffers; b) Processing of case II in 5.2.3.1: When there is no video in the cache, the audio pts are used as the calculation point, from the last monotonically increasing audio frame to the latest AudioPts All audio frames are used as valid caches.

5.3. LAS Request Logic Processing Specification

The LAS request logic processing specification describes the specific response logic after the server receives the LAS request.

5.3.1. In LAS requests, the optional @audioOnly Default or @audioOnly=false at the same time

@startPts Default

Use the default value of @startPts (@startPts = defaultStartPts) to send the laS client the video and audio streams specified in the @url of the LAS request, as follows:

I． When there is video in the cache: pts is closest to latestVideoPts from the currently valid buffer

- |defaultStartPts| The I frame starts sending the media stream

II． When there is no video in the cache: pts closest from the current valid cache

latestAudioPts - |defaultStartPts| The audio frame starts sending a media stream

5.3.2. In LAS requests, the optional @audioOnly=true while @startPts default

Use the default value for @startPts (@startPts = defaultStartPts) to send the @url specified audio stream in the LAS request to the LAS client as follows:

I. pts is closest to latestAudioPts from the currently valid buffer - |faultStartPts |

The audio frame starts sending a media stream

5.3.3. In a LAS request, the optional @audioOnly Default or @audioOnly=false , at the same time

@startPts=0

Use @startPts=0 to send the laS client the video and audio streams specified in the @url of the LAS request, as follows:

I. When there is video in the cache: pts is closest to latestVideoPts from the currently valid buffer

The I frame starts sending the media stream

II. When there is no video in the cache: pts is closest to latestAudioPts from the current valid buffer

The audio frame starts sending a media stream

5.3.4. In a LAS request, the optional @audioOnly=true and the @startPts=0

Use @startPts=0 to send the @url specified audio stream in the LAS request to the LAS client with the following rules:

I. Sends a media stream starting from the audio frame in the current valid buffer where pts is closest to latestAudioPts

5.3.5. In a LAS request, the optional @audioOnly Missing or @audioOnly=false , at the same time

@startPts<0

Using @startPts<0, send the laS client the video and audio streams specified in the @url in the LAS request, as follows:

I. When there is video in the cache: pts is closest to latestVideoPts from the currently valid buffer

- |@startPts| The I frame starts sending the media stream

II. When there is no video in the cache: pts closest from the current active hold

latestAudioPts - |@startPts| The audio frame starts sending a media stream

5.3.6. In a LAS request, the optional @audioOnly=true and @startPts<0

Use @startPts<0 to send the @url specified audio stream in the LAS request to the LAS client with the following rules:

I. pts is closest to latestAudioPts - from the currently valid buffer - |@startPts| Audio

The frame starts sending a media stream

5.3.7. In a LAS request, the optional @audioOnly Default or @audioOnly=false at the same time

@startPts>0

Using @startPts>0, send the LAS client the video and audio streams specified in the @url request, as follows:

I. When there is a video in the cache and there is no timestamp fallback: Starts with the smallest I frame of pts , along

pts increases in the direction in which the I frame that finds the maximum pts and satisfies the pts < = @startPts starts sending media streams; if the pts are not found < = @startPts I frame, the first one is found

pts meets the > @startPts I frames to start sending media streams; if none are found, wait

Wait for the first I frame of pts greater than or equal to @startPts to arrive before starting to send media streams

II. When there is video in the cache and there is a timestamp fallback, the media stream is sent starting with the latest I frame

III. When there is no video in the cache and no timestamp fallback: starting with the smallest audio frame of pts , in the direction of pts increment, Find the first pts > = @startPts audio frame to start sending media streams, if you can't find an audio frame that meets the pts >= @startPts , Then wait for the first one

pts greater than or equal to @startPts audio frames or I frames arrive before starting to send media streams

IV. When there is no video in the cache and there is a timestamp fallback, the media stream is sent starting from the latest audio frame

5.3.8. In a LAS request, the optional @audioOnly=true and @startPts>0

Using @startPts>0, send the LAS client the audio stream specified in the @url in the LAS request with the following rules:

I. When there is no timestamp fallback: Starting with the smallest audio frame of pts , along the direction where pts increases, find the first one /b20> The audio frame of pts >= @startPts starts sending a media stream, if you can't find an audio frame that satisfies the pts >= @startPts , then wait for the first pts to be greater than or equal to

@startPts the audio frame arrives before starting to send the media stream

II. When there is a timestamp fallback, the media stream is sent starting from the most recent audio frame

5.3.9. Server-side internal back-to-source

When the edge node does not have the media stream specified by the LAS request, it needs to pull the media stream back to the source to the parent node:

I. Third-party origins must carry the @startPts field back to the source

II. The internal node returns to the source, and if it carries @startPts field, press LAS request, based on

The actual value of the @startPts is returned to the source, otherwise it is tacitly recognized

@startPts=defaultStartPts

5.3.10. Exception Handling

When @startPts>0, if a media frame with pts or less than or equal to @startPts does not exist in the valid buffer, there are two ways to handle it:

I. Wait modes such as 5.3.7 and 5.3.8

II. Error Handling Mode: When @startPts exceeds a certain threshold of the maximum pts in the valid buffer

(timeout threshold timeoutPts), the decision is that the LAS request is an error. Threshold timeoutPts

Supported configuration

6. LAS client description

The specific implementation of the LAS client is not in the scope of the LAS standard. LAS only gives recommended implementation architectures and adaptive algorithm strategies.

6.1. LAS Client Architecture

As shown in the figure, the main logic of the client includes:

I. MPD parsing: Responsible for parsing MPD and obtaining corresponding media information, such as each media representation

URL, bitrate, id, and other information

II. Downloader: Responsible for downloading the media stream and passing the media data to the decoded rendering module, collecting network state information and passing it to the Adaptive Policy module

III. Decoding rendering: Receives media data from the downloader and decodes the rendering, while passing information such as playback-related states such as buffer size, stuttering, dropped frames, etc., to the adaptive policy module

IV. Adaptive strategy: Combined with the alternate media representation set obtained by MPD parsing, the network status transmitted by the downloader, and the playback status of the decoding rendering module feedback, the best media representation is comprehensively determined and passed to the downloader for download switching of the media representation.

|  |
| --- |
|  |
|  |  |

Figure 6.1 LAS client diagram

6.2. MPD Parsing

It is mainly responsible for parsing the media rendering description file and passing each element and attribute to the corresponding module.

6.3 . Downloader

Based on the relevant information obtained from the adaptive policy module on the media representation, a LAS request is generated and sent to the server. In LAS , media representation is streamed, and requests need to be resend only when a media representation switch occurs.

6.3.1. LAS Request Generation

6.3.1.1. Initiating LAS Requests

When starting, according to the initial media representation specified by the business or the default initial media representation in MPD, obtain the corresponding @url of the media representation to be requested, record: startUrl.

Set the @startPts to a negative number, such as -8000, which means that it is expected to pull 8 seconds of cached data, that is, the client buffer has up to 8 seconds of buffered data, achieving a compromise between latency and network jitter buffering The specific value is specified by the client business party, for example, it is specified as initStartPts. As shown in Figure 7.2, where blue represents the keyframe, when set @startPts = -8000, the data is actually pulled to 10000ms (pts = 12000 ms ~ pts = ). 22000𝑚𝑠）。

Therefore, the request for a started LAS is: startUrl & startPts=initStartPts.

|  |
| --- |
|  |
|  |  |

Figure 6.2 Schematic diagram of the start @startPts

6.3.1.2. Media represents a LAS request for switching

I. During media playback, if the media representation of the adaptive policy output matches the media representation that is currently being downloaded, the output of the current adaptive policy is ignored without generating a new LAS request or sending the request

II. During media playback, if the media representation of the adaptive policy output is inconsistent with the media representation currently being downloaded, a) obtains the corresponding id according to the media representation of the adaptive policy output @url, denoted as switchUrl; b) the pts of the keyframe represented by the media output according to the adaptive policy, are obtained rtPts, denoted as switchedStartPts.

Therefore, the request for a started LAS is: switchUrl & startPts=switchedStartPts.

6.3.2. Network Status Collection

While the media is downloaded, it is responsible for collecting the network status. Unlike the traditional shard-based request/download model, in LAS, streaming is used. At the network state collection level, the mode of fixed time collection point is adopted, that is, every fixed time T(ms) is used to calculate the actual amount of data downloaded during this time period

S(Bytes), resulting in a bandwidth sample point𝐵(𝑘𝑏𝑝𝑠) = 𝑆 ∗ 8/𝑇。 Typically, T = 500ms. nt.

Note: The reference implementation logic for bandwidth estimation will be given in the reference code

6.4 . Decode the rendering

6.4.1. Decoding Rendering

When decoding rendering, the principle of high quality is the principle, that is, when the bitrate switch occurs, if the high and low bitrates overlap, the media representation of the high bitrate is played first. In addition, when switching between high and low bitrate rendering, it is aligned according to pts for seamless switching.

6.4.2. Playback Status Collection

When the player decodes and renders, it passes the player's state information to the adaptive policy module at regular intervals. It is recommended here to be consistent with the time interval T in 7.3.2, that is, to trigger with the same timer. The corresponding status information includes the current video cache size, the current audio cache size, the last lag time, the last caton time, the drop rate in T(ms), and so on.

6.5 . Adaptive policies

The adaptive strategy combines network status information and playback status information to dynamically select the best media representation, achieving a compromise between stutter rate, clarity, and smoothness. Two switching schemes are recommended here: GOP boundary decisions and arbitrary point decisions.

6.5.1. GOP Boundary Decisions

In LAS, media is streaming-based, and LAS clients pass through header information (for example

flv\_tag\_header) to get the type of frame being transmitted. If it is an I frame, it means that the previous GOP download is over, and the adaptive policy is triggered to make a bitrate adaptive decision to determine the media representation of the next GOP.

Specifically, when the LAS client obtains the first frame of the i + 1 GOP through the header information as the I frame, it determines the ith frame GOP has been downloaded, and the adaptive decision strategy is triggered to make a bitrate adaptive decision to determine the id of the best media representation of the i + 1st GOP /b118>and pass its corresponding URL and the pts of the first frame of the i + 1 GOP to the download module and generate it LAS request, as shown in the following illustration, where blue represents keyframes (I frames).

|  |
| --- |
|  |
|  |  |

Figure 6.3 Schematic diagram of @startPts based on GOP boundary switching

6.5.2. Arbitrary Point Decision Making

Arbitrary decision-making means that the adaptive policy can be triggered at any time, without waiting for the current GOP

End of load.

For ease of description, defines the set of bitrates for media descriptions as < r<, 𝑟>, ... , 𝑟@ >(𝑘𝑏𝑝𝑠), the bitrate represented by the currently downloaded media is rB, the length of the GOP 𝐷(𝑚𝑠), through the sampling point, the estimated final reliable bandwidth is BDEF(𝑘𝑏𝑝𝑠), the current GOP has been downloaded𝑑(𝑚𝑠)。 Based on the dual threshold model, the LAS client presets two cache thresholds qIand qJ, represents when the amount of media cache is greater than qIWhen playback is in a safe state, the probability of stuttering is very small, and you can consider increasing the bitrate of the media representation to improve the clarity; on the contrary, when the media cache amount is less than qJWhen playback is in a dangerous state, a stutter occurs

The rate is very large, and it is necessary to consider reducing the bit rate of media representation to avoid stuttering.

Considering the reference relationship of decoding, when a media representation switch occurs, it must be opened from the first frame of the GOP

Download first. Suppose the bitrate of the media representation that actually needs to be downloaded is r:

I. When r = rB, the download continues from the current GOP , and the amount of cached data (duration) at the end of the GOP download is

𝑞 = 𝑞B + 𝐷 − 𝑑 − (𝐷 − 𝑑) ∗ 𝑟B  ∗ 8/𝐵DEF (1)

II. When r! = rB, the download needs to start with the first frame of the GOP and end at the end of the GOP download

The amount of cached data (duration) is

𝑞 = 𝑞B + 𝐷 − 𝑑 − 𝐷 ∗ 𝑟 ∗ 8/𝐵DEF (2)

6.5.2.1. The current amount of media cache is greater than qI

Try increasing the bitrate of the media representation to improve clarity. The basic principle is that selecting a media indicates a download, and at the end of the current GOP download, the media cache is not less than qIUnder the premise, select the media representation with the largest bitrate, specifically:

I. In Equation (2), if for any r > rB , no q > q existIOf the media said,

Then the bitrate represented by the media is kept unchanged, that is, the bitrate of the continued download is r = rBThe media said

II. In Equation (2), if for any r > rB , if q > qIOf the media said,

Then select Q > QIAnd the media with the largest bitrate indicates that a request for download is made. and set

@startPts is the pts for the first frame of the GOP, that is, the first frame of the GOP is requested for download

6.5.2.2. Media cache is less than qJ

It is necessary to reduce the bit rate of media representations and avoid stuttering. The basic principle is that selecting a media representation for download and at the end of the current GOP download, the media cache is not less than qJUnder the premise, select the media representation with the largest bitrate, specifically:

I. In Equation (2), if for any r, there is no q ≥ qJThe media said that it was the most

The final requested media indicates the bit rate:

𝑟∗ = 𝑎𝑟𝑔𝑚𝑎𝑥{(1), (2)}

If r∗ = 𝑟B, the download of the current media continues without sending a request; otherwise, the download bitrate is equal to r∗The media representation and settings @startPts pts for the first frame of the GOP , that is, the request for download starting from the first frame of the GOP

In Equation (2), if there is q for any r, if q ≥ qJThe media representation, then select q ≥ qJ And the media with the largest bitrate indicates that a request for download is made. and set

@startPts is the pts for the first frame of the GOP, that is, the first frame of the GOP is requested for download.

Figure 6.4 depicts the relationship between the value of the @startPts and whether the bitrate switches in any point switching.

Figure 6.4 Schematic diagram of switching @startPts based on any point

|  |
| --- |
|  |
|  |  |

Appendix A. MPD Examples

|  |
| --- |
|  |
|  |  |

Appendix B. Examples of LAS Requests

[](http://www.bing.com/translator)

**Original**

LAS只有在起播或发生码率切换时才需要发送请求

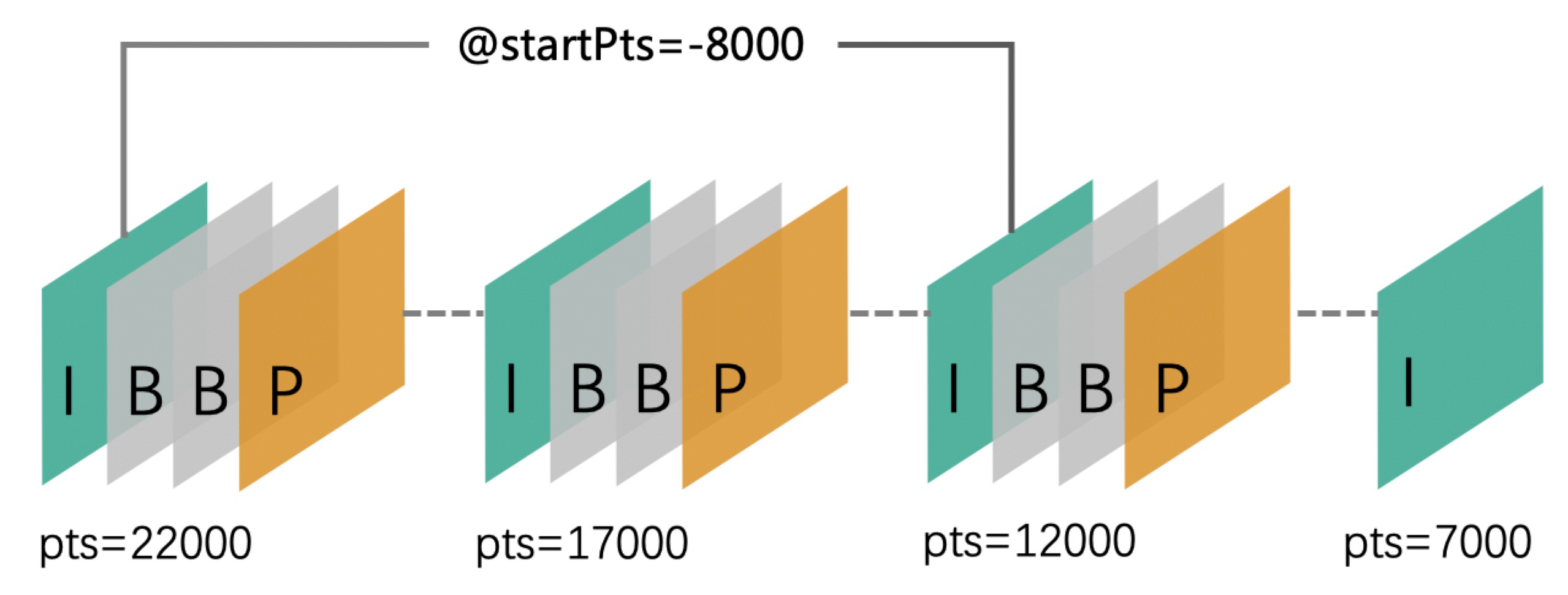


图 6.2 启播@startPts 示意图

* + - 1. 媒体表示切换的 LAS 请求
         1. 在媒体播放过程中，如果自适应策略输出的媒体表示与当前正在下载的媒体表示一致，则忽略当前自适应策略的输出，不用生成新的 LAS 请求，也不用发送请求
         2. 在媒体播放过程中，如果自适应策略输出的媒体表示与当前正在下载的媒体表示不一致，则 a）依据自适应策略输出的媒体表示的 id，获取对应的@url，记为 switchUrl；b）依据自适应策略输出的媒体表示的关键帧的 pts，获取@startPts，记为 switchedStartPts。

因此，启播的 LAS 的请求为：switchUrl&startPts=switchedStartPts。

* + 1. 网络状态收集

在媒体下载的同时，负责收集网络状态。与传统的基于分片请求/下载的模式不同，在 LAS 中，采用流式传输。在网络状态收集层面，采用固定时间采点的模式，即每隔一个固定时间𝑇(𝑚𝑠)，统计该时间段实际下载的数据量

𝑆(𝐵𝑦𝑡𝑒𝑠)，从而得到一个带宽采样点𝐵(𝑘𝑏𝑝𝑠) = 𝑆 ∗ 8/𝑇。典型的，𝑇 = 500𝑚𝑠。基于这些带宽的采样点，通过滤波和预测算法，估计网络的真实带宽，作为 码率调整的依据。

注：带宽估计的参考实现逻辑，将在参考代码给出

* 1. .解码渲染
     1. 解码渲染

解码渲染时，以高质量优先为原则，即当发生码率切换时，如果高低码率存在重叠的情况，则优先播放高码率的媒体表示。此外，在高低码率渲染切换时，依据 pts 进行对齐，从而达到无缝切换。

* + 1. 播放状态收集

播放器在解码渲染时，每隔一定时间间隔，将播放器的状态信息传递给自适应策略模块。这里推荐与 7.3.2 中的时间间隔 T 一致，即采用同一个定时器触发。相应的状态信息包括当前视频缓存大小、当前音频缓存大小、最近一次卡顿时间、最近一次卡顿时长、𝑇(𝑚𝑠)内的丢帧率等。

* 1. .自适应策略

自适应策略结合网络状态信息和播放状态信息，动态选择最佳的媒体表示， 在卡顿率、清晰度、平滑性之间取得折中。这里推荐两种切换方案： GOP 边界决策和任意点决策。

* + 1. GOP 边界决策

在 LAS 中，媒体是基于流式传输的， LAS 客户端通过头信息（例如

flv\_tag\_header）来获取当前正在传输的帧的类型。如果为 I 帧，则说明上一个GOP下载结束，此时触发自适应策略做码率自适应决策，决定下一个GOP 的媒体表示。

具体而言，当 LAS 客户端通过头信息获取到第𝑖 + 1个GOP 的第一帧为 I 帧时，就判定第𝑖个 GOP 已经下载完成，此时便触发自适应决策略做码率自适应决策，确定第𝑖 + 1个 GOP 的最佳媒体表示的 id，并将其对应的 url 以及第𝑖 + 1个GOP 的第一帧的 pts 传递给下载模块，并生成 LAS 请求，如下图所示，其中蓝色代表关键帧（I 帧）。

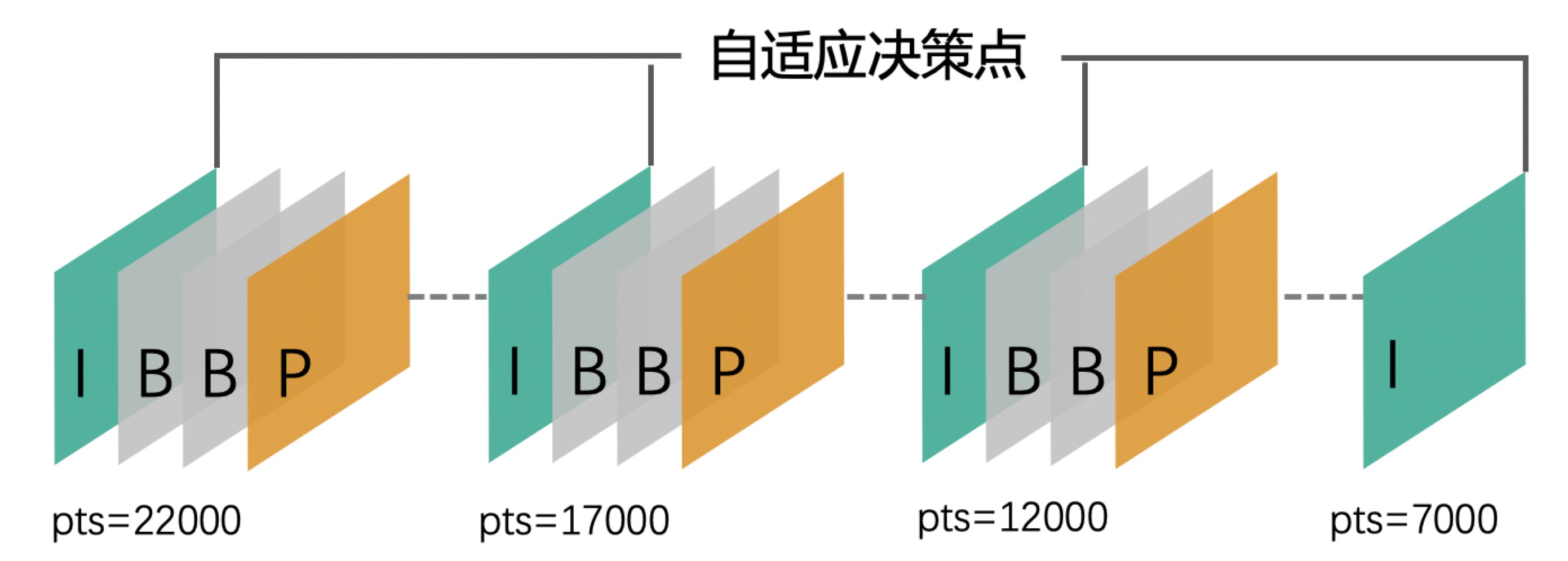


图 6.3 基于 GOP 边界切换@startPts 示意图

* + 1. 任意点决策

任意点决策是指自适应策略可以在任意时刻触发，而不用等到当前 GOP 下

载结束。

方便描述，定义媒体描述的码率集合为< 𝑟<, 𝑟>, … , 𝑟@ > (𝑘𝑏𝑝𝑠)，当前下载的媒体表示的码率为𝑟B，GOP 的长度𝐷(𝑚𝑠)，通过采样点，估计出的最终可靠的带宽为𝐵DEF(𝑘𝑏𝑝𝑠)，当前 GOP 已经下载完𝑑(𝑚𝑠)。基于双阈值模型，LAS 客户端预设定两个缓存阈值𝑞I和𝑞J，分别代表当媒体缓存量大于𝑞I时，播放处于安全状态，发生卡顿的概率很小，可以考虑增大媒体表示的码率，提升清晰度；相反，当媒体缓存量小于𝑞J时，播放处于危险状态，发生卡顿的概

率很大，需要考虑降低媒体表示的码率，避免卡顿。

考虑到解码的参考关系，当发生媒体表示切换时，必须从 GOP 的第一帧开

始下载。假设实际需要进行下载的媒体表示的码率为𝑟，则:

1. 当𝑟 = 𝑟B，从当前 GOP 继续下载，并且 GOP 下载结束时的缓存数据量（时长）为

𝑞 = 𝑞B + 𝐷 − 𝑑 − (𝐷 − 𝑑) ∗ 𝑟B ∗ 8/𝐵DEF (1)

1. 当𝑟! = 𝑟B，需要从 GOP 的第一帧开始下载，并且 GOP 下载结束时的

缓存数据量（时长）为

𝑞 = 𝑞B + 𝐷 − 𝑑 − 𝐷 ∗ 𝑟 ∗ 8/𝐵DEF (2)

* + - 1. 当前媒体缓存量大于𝑞I

尝试增大媒体表示的码率，提升清晰度。基本原则为，选择一个媒体表示进行下载，并且在当前 GOP 下载结束时，在媒体缓存不低于𝑞I的前提下，选择码率最大的媒体表示，具体而言：

1. 在公式（2）中，如果对于任意的𝑟 > 𝑟B ，不存在𝑞 > 𝑞I的媒体表示，

则保持媒体表示的码率不变，即继续下载码率为𝑟 = 𝑟B的媒体表示

1. 在公式（2）中，如果对于任意的𝑟 > 𝑟B ，若存在𝑞 > 𝑞I的媒体表示，

则选择满足𝑞 > 𝑞I且码率最大的媒体表示进行请求下载。并且设置

@startPts 为该 GOP 第一帧的 pts，即从该 GOP 的第一帧开始请求下载

* + - 1. 媒体缓存量小于𝑞J

需要降低媒体表示的码率，避免卡顿。基本原则为，选择一个媒体表示进行下载，并且在当前 GOP 下载结束时，在媒体缓存不低于𝑞J的前提下，选择码率最大的媒体表示，具体而言：

I. 在公式（2）中，如果对于任意的𝑟，不存在𝑞 ≥ 𝑞J的媒体表示，则最

终请求的媒体表示的码率为：

𝑟∗ = 𝑎𝑟𝑔𝑚𝑎𝑥{(1), (2)}

若𝑟∗ = 𝑟B，则继续下载当前媒体表示，不用发送请求；否则，下载码率等于𝑟∗的媒体表示，并且设置@startPts 为该 GOP 第一帧的 pts， 即从该 GOP 的第一帧开始请求下载

在公式（2）中，如果对于任意的𝑟，若存在𝑞 ≥ 𝑞J的媒体表示，则选择满足𝑞 ≥ 𝑞J 且码率最大的媒体表示进行请求下载。并且设置

@startPts 为该 GOP 第一帧的 pts，即从该 GOP 的第一帧开始请求下载。

图 6.4 描述了在任意点切换中，@startPts 的取值与码率是否切换的关系。

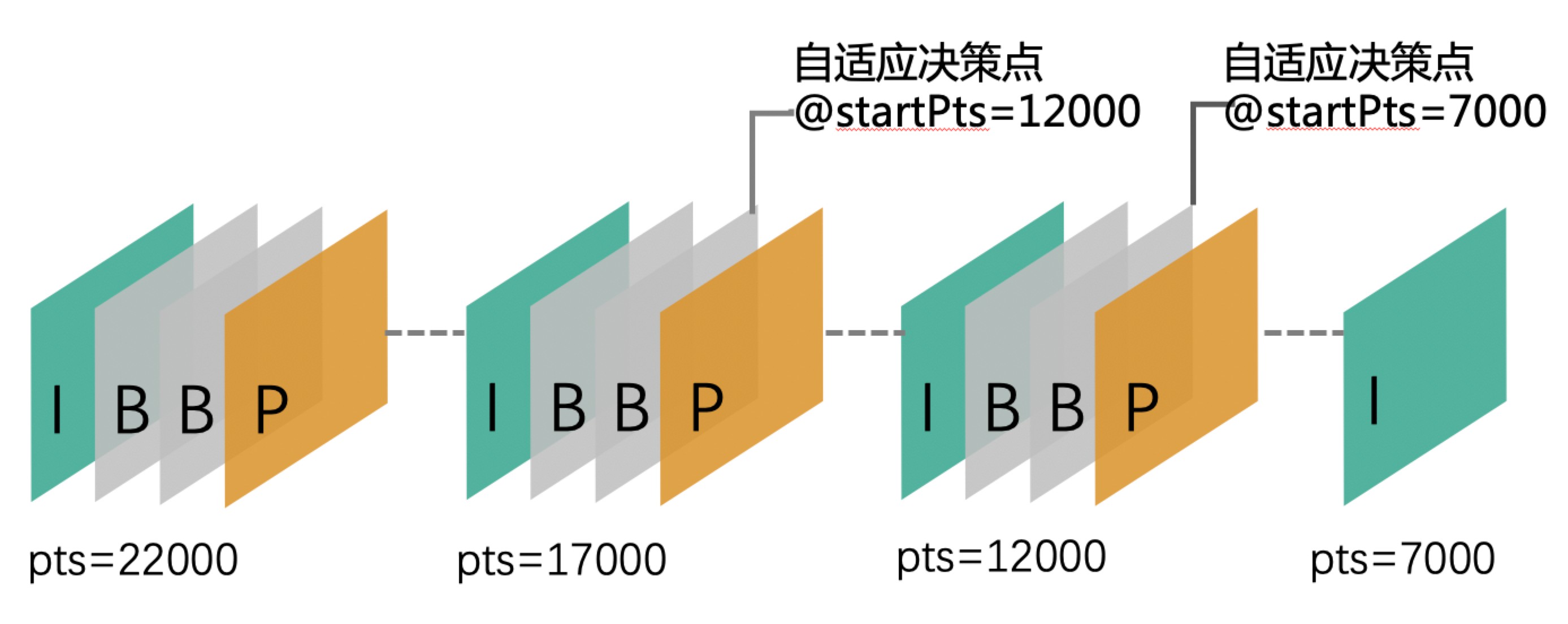


图 6.4 基于任意点切换@startPts 示意图

附录 A. MPD 示例

{

"version": "1.0.0", "adaptationSet": [

{

"duration": 1000,

"id": 1,

"representation": [

{

"id": 1,

"codec": "avc1.64001e,mp4a.40.5",

"url": "https://las-tech.org.cn/kwai/las-

test\_ld500d.flv",

"backupUrl": [],

"host": "las-tech.org.cn", "maxBitrate": 700,

"width": 640,

"height": 360,

"frameRate": 25, "qualityType": "SMOOTH", "qualityTypeName": " 流 畅 ", "hidden": false, "disabledFromAdaptive": false, "defaultSelected": true

},

{

"id": 2,

"codec": "avc1.64001f,mp4a.40.5",

"url": "https://las-tech.org.cn/kwai/las-

test\_sd1000d.flv",

"backupUrl": [],

"host": "las-tech.org.cn",

"maxBitrate": 1300,

"width": 960,

"height": 540,

"frameRate": 25, "qualityType": "STANDARD", "qualityTypeName": " 标 清 ", "hidden": false, "disabledFromAdaptive": false, "defaultSelected": false

},

{

"id": 3,

"codec": "avc1.64001f,mp4a.40.5",

"url": "https://las-tech.org.cn/kwai/las-test.flv", "backupUrl": [],

"host": "las-tech.org.cn", "maxBitrate": 2300,

"width": 1280,

"height": 720,

"frameRate": 30, "qualityType": "HIGH", "qualityTypeName": " 高 清 ", "hidden": false, "disabledFromAdaptive": false, "defaultSelected": false

}

]

}

]

}

附录 B. LAS 请求示例

https://las-tech.org.cn/kwai/las-test.flv

https://las-tech.org.cn/kwai/las-test.flv?audioOnly=false https://las-tech.org.cn/kwai/las-test.flv?audioOnly=true https://las-tech.org.cn/kwai/las-test.flv?startPts=0 https://las-tech.org.cn/kwai/las-test.flv?startPts=5678536 https://las-tech.org.cn/kwai/las-test.flv?startPts=-7000

https://las-tech.org.cn/kwai/las-test.flv?audioOnly=false&startPts=0 https://las-tech.org.cn/kwai/las-test.flv?audioOnly=false&startPts=5678536 https://las-tech.org.cn/kwai/las-test.flv?audioOnly=false&startPts=-7000 https://las-tech.org.cn/kwai/las-test.flv?audioOnly=true&startPts=0 https://las-tech.org.cn/kwai/las-test.flv?audioOnly=true&startPts=5678536 https://las-tech.org.cn/kwai/las-test.flv?audioOnly=true&startPts=-7000