

# Matroska解封装原理与实践

Matroska Demuxing Principle and Practice



哔哩哔哩

王妍君

2024年5月

# 目录

## Contents

### 01. 背景

01. Background

### 02. 原理

02. Principle

### 03. 方案

03. Scheme

### 04. 应用

04. Application

### 05. 总结与展望

05. Conclusion and Outlook

# 01. 背景

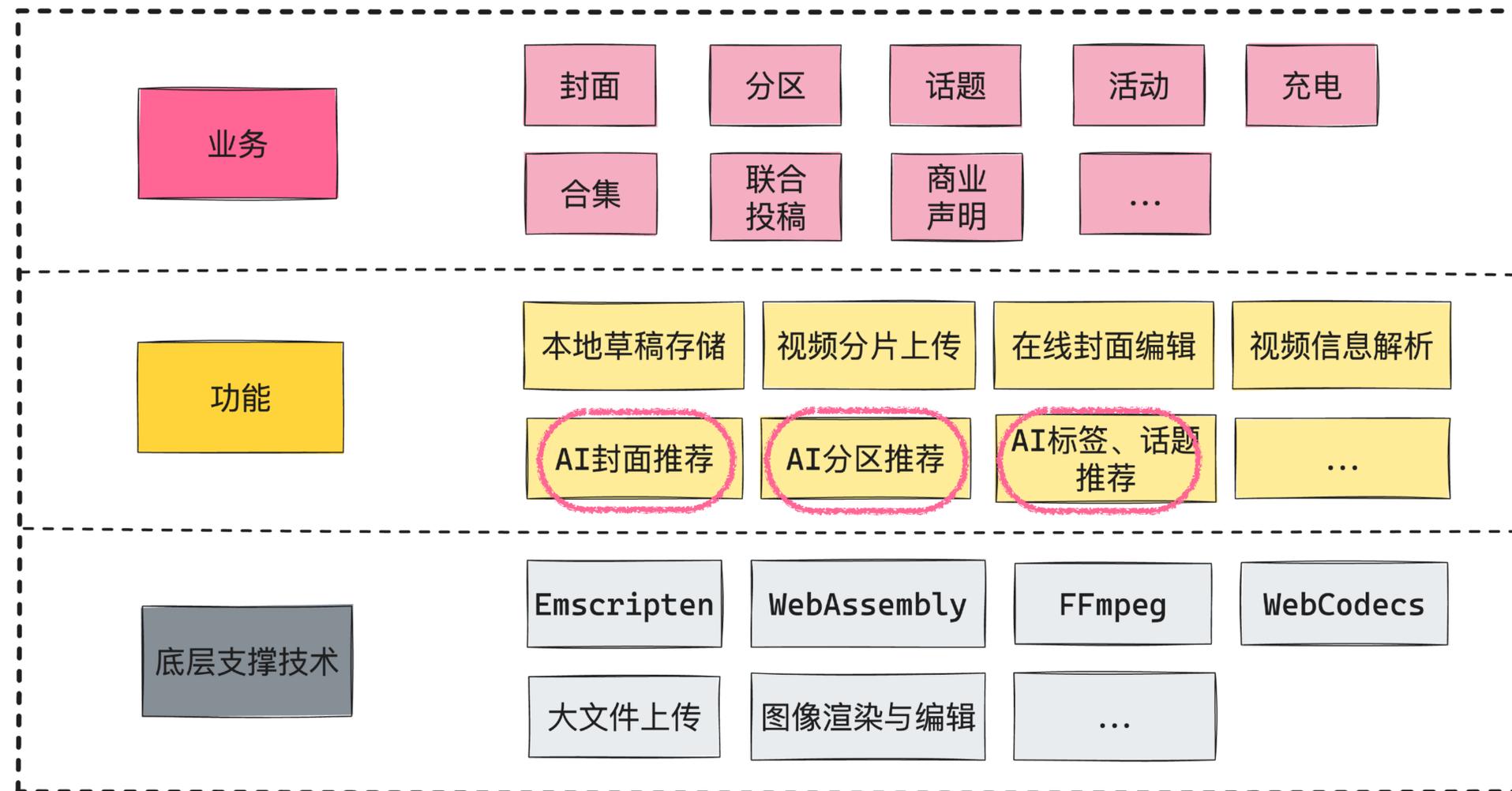
01. Background

# 01. 背景

## B站Web投稿页：

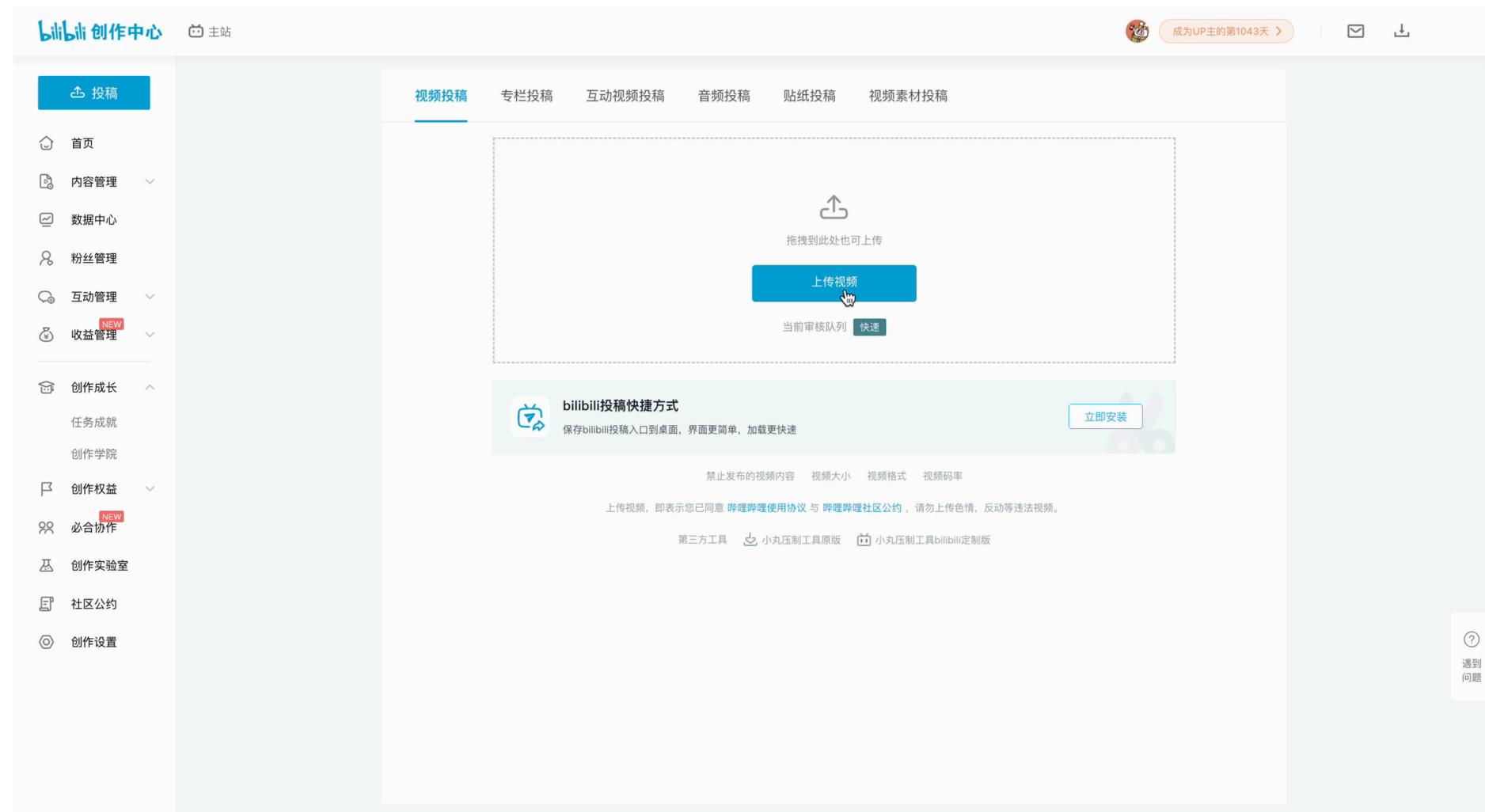
B站UGC内容生产重要入口

在所有端投稿量中占比最高



# 01. 背景

## B站Web投稿页



## 推荐封面



## 推荐分区



## 推荐标签

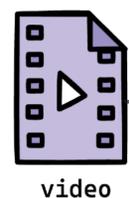


前端来完成视频的解析，获取视频截帧图片，基于图片进行计算

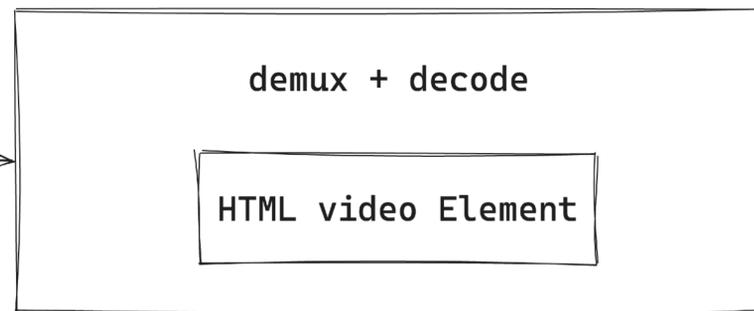
# 01. 背景

## 视频截帧画面获取

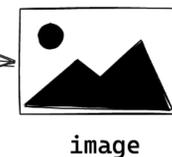
基础方案



输入



输出

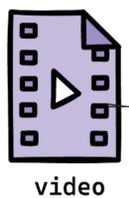


优点：简单易用、兼容性好

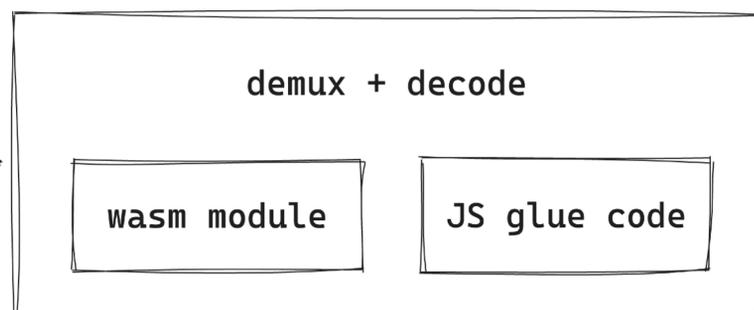
缺点：支持格式有限

通用方案

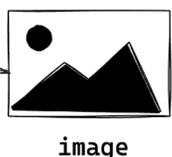
降级



输入



输出

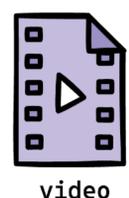


优点：支持几乎所有音视频格式、兼容性好

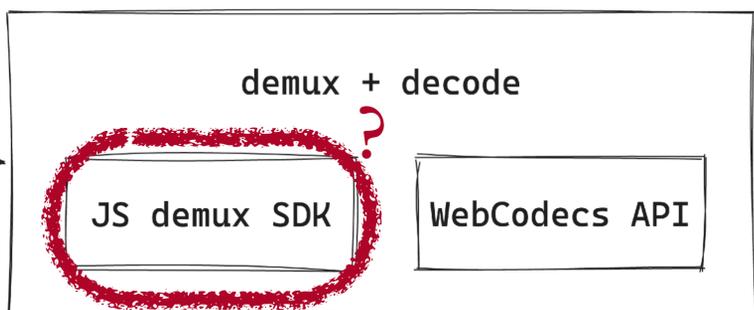
缺点：性能和资源消耗大

升级方案

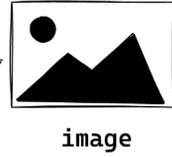
降级



输入



输出



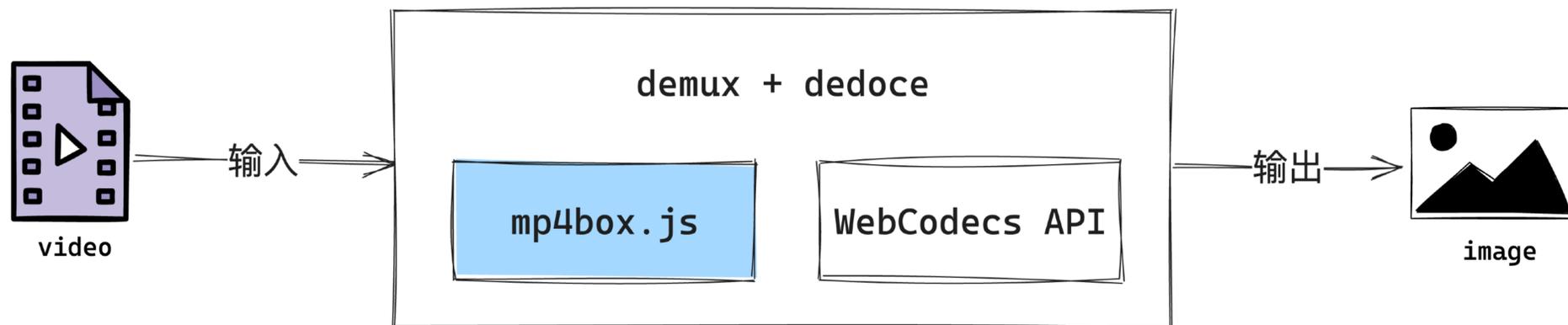
优点：高效利用硬件加速，解码效率高

缺点：兼容性有限

# 01. 背景

升级方案:

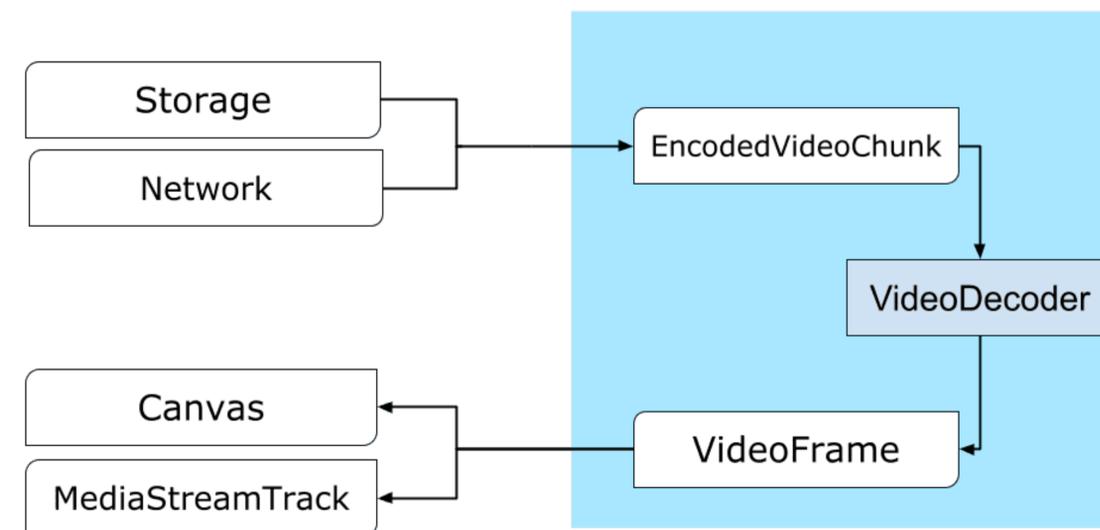
mp4视频



mp4box.js

```
{  "track_id":4,  "description": "[Box]",  "is_rap":true,  "timescale":1000,  "dts":0,  "cts":0,  "duration":1000,  "size":41,  "data": "[ArrayBuffer]"}
```

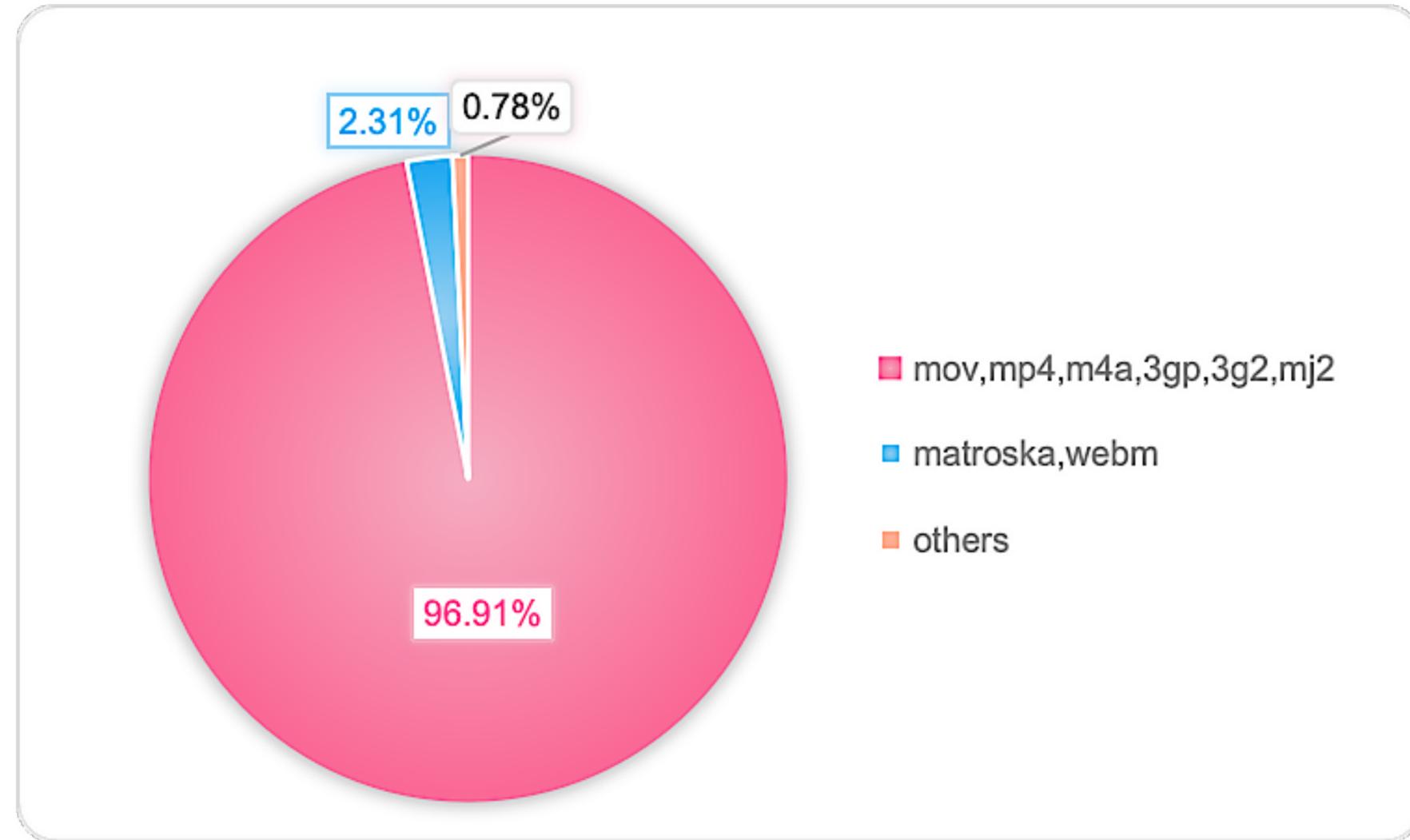
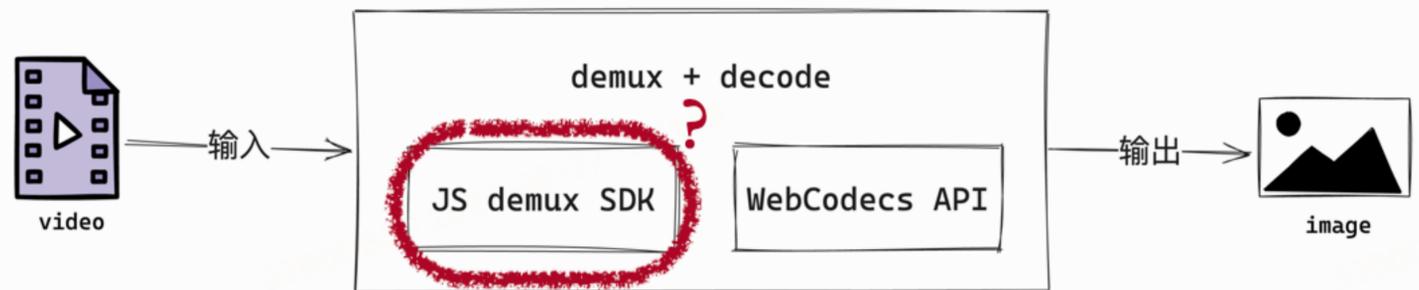
WebCodecs API



# 01. 背景

- Matroska、WebM视频占比超过2%，是除MP4以外占比最高的格式
- 在各粉丝段都存在较为固定的UP主群体

Matroska、WebM  
JS demux SDK?



# 02. 原理

02.Principle

# 02. 原理 MATROŠKA

- 一种多媒体封装格式
  - 常见扩展名有 .mkv、.mka 等
- 灵活、强大
  - 可支持任意编码的音频、视频，可支持任意格式的字幕
  - 可容纳多视频轨，多音轨，多字幕
  - 可添加附件、标签、章节等信息
  - 兼容性差

可添加多字幕，可添加附件

| Type                  | Codec/MIME type  | Language           |
|-----------------------|------------------|--------------------|
| > Segment information |                  |                    |
| > Video track 1       | V_MPEGH/ISO/HEVC | Undetermined (...) |
| > Audio track 2       | A_FLAC           | Japanese (ja)      |
| > Subtitle track 3    | S_TEXT/ASS       | Chinese (zh)       |
| > Subtitle track 4    | S_TEXT/ASS       | Chinese (zh)       |
| ▼ Attachments         |                  |                    |
| NIWONIPA Regular.ttf  | font/ttf         |                    |

可添加章节

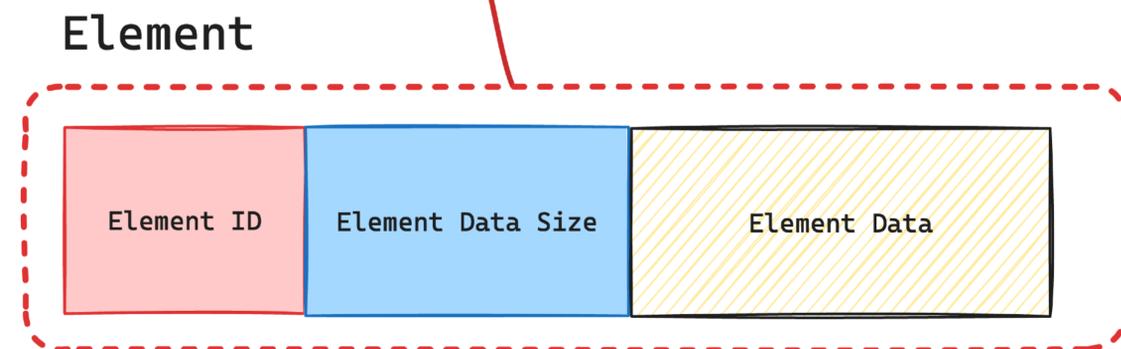
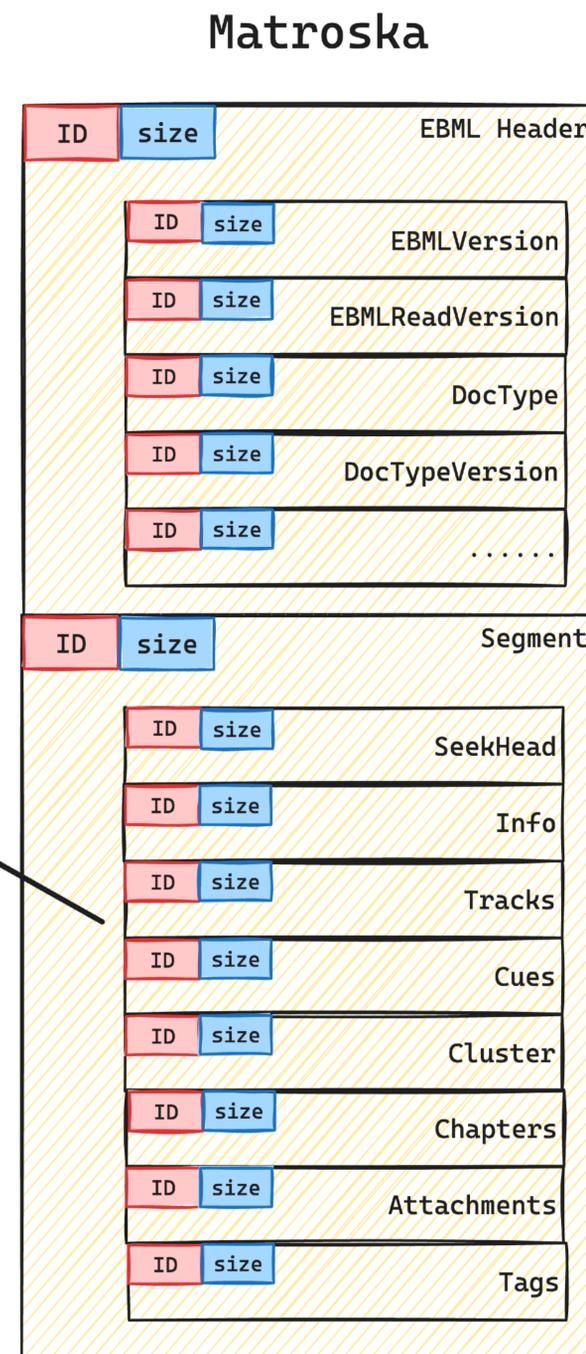
| Edition/Chapter | Start              | End                |
|-----------------|--------------------|--------------------|
| ▼ Edition entry |                    |                    |
| Chapter 01      | 00:00:00.000000000 | 00:00:56.056000000 |
| Chapter 02      | 00:00:56.056000000 | 00:02:26.021000000 |
| Chapter 03      | 00:02:26.021000000 | 00:13:14.001000000 |
| Chapter 04      | 00:13:14.001000000 | 00:22:07.743000000 |
| Chapter 05      | 00:22:07.743000000 | 00:23:42.035000000 |



# 02. 原理

- 由一系列Element组成
- 主要包含EBML Header、Segment两大Element

|   |
|---|
| <b>SeekHead</b><br>包含了其他Element的位置，帮助在文件中快速查找内容 |
| <b>Info</b><br>包含关于文件的元数据信息，如标题、作者、日期等          |
| <b>Tracks</b><br>描述文件中包含的音视频轨道的信息               |
| <b>Cues</b><br>包含可快速定位关键帧Cluster的索引信息           |
| <b>Cluster</b><br>包含多媒体数据和时间跨度，是实际存储音视频数据的地方    |
| <b>Chapters</b><br>包含章节信息，用于导航和定位               |
| <b>Attachments</b><br>用于存储与文件相关的附加文件，如字体文件等     |
| <b>Tags</b><br>包含用户定义的标签和注释                     |





# 03. 方案

03. Scheme

# 03. 方案

## 目标

- 遵循Matroska的结构，使用JavaScript对Matroska视频进行解封装

## 要求

- 内存占用小
- 解析耗时短，解析效率高
- 可获取视频元信息
- 可获取特定时间戳的视频帧数据，结合WebCodecs API进行解码

## 调研

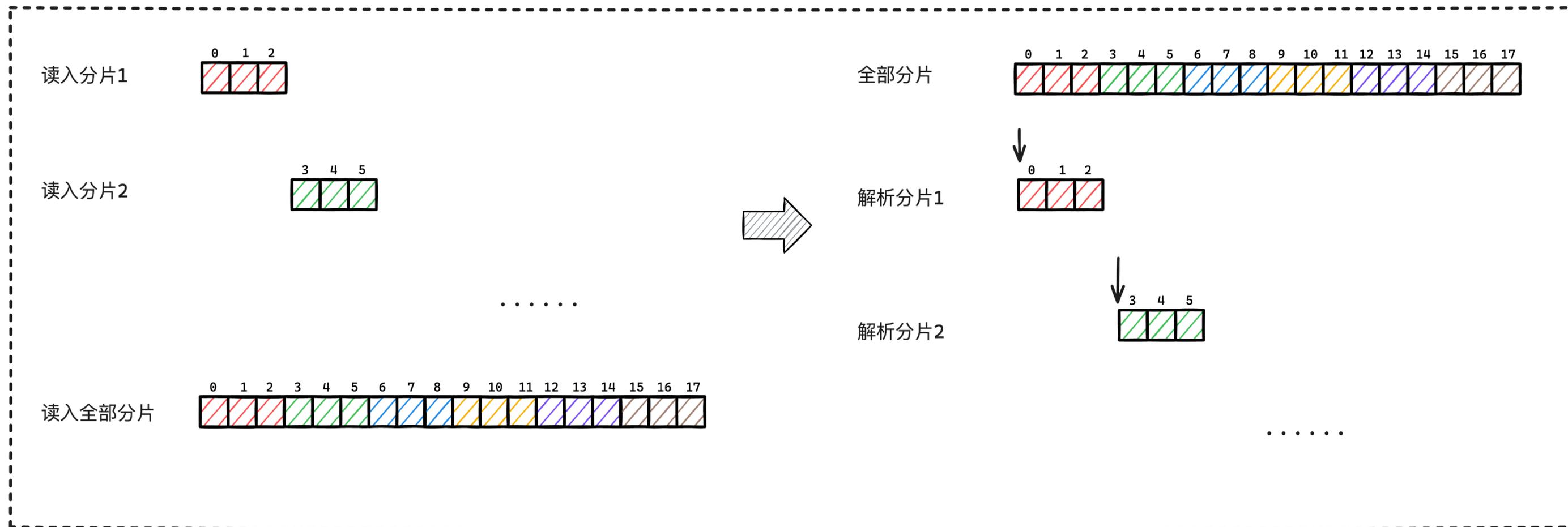
- jswebm:
  - <https://www.npmjs.com/package/jswebm>
  - 内存占用大，解析耗时久，错误处理不完善
- ebml:
  - <https://www.npmjs.com/package/ebml>
  - 无法按需解析，需要二次解析，运行在node环境

# 03. 方案 常见文件读取方式

- 读入：将文件转换为ArrayBuffer，切片后分片传入或整个传入
- 使用：对文件分片按顺序读取



只能按顺序读  
内存占用大

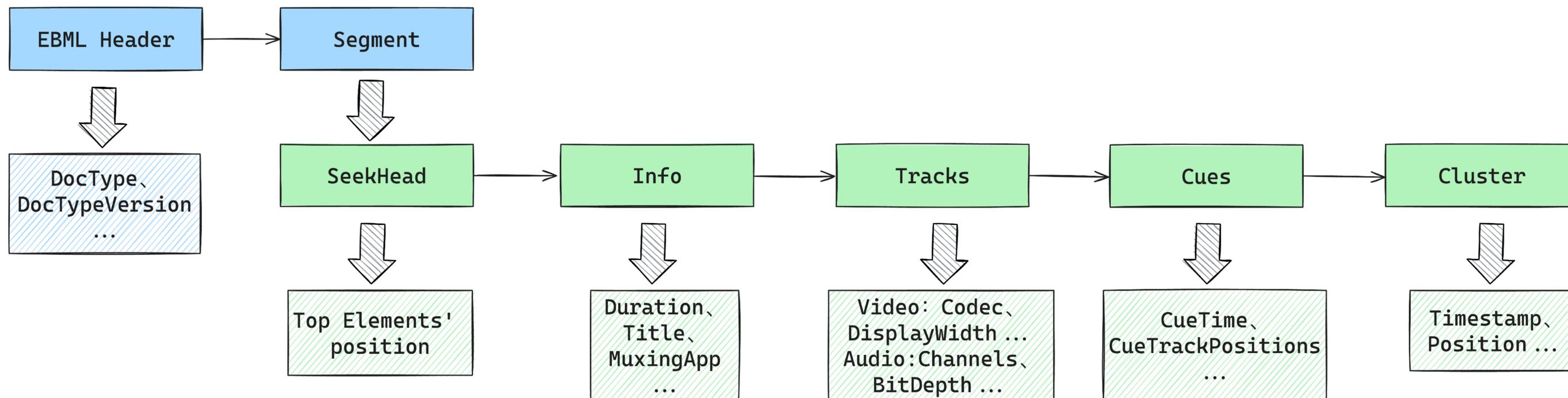


# 03. 方案 常见内容解析方式

- 按顺序递归展开解析
- 一次性全部解析

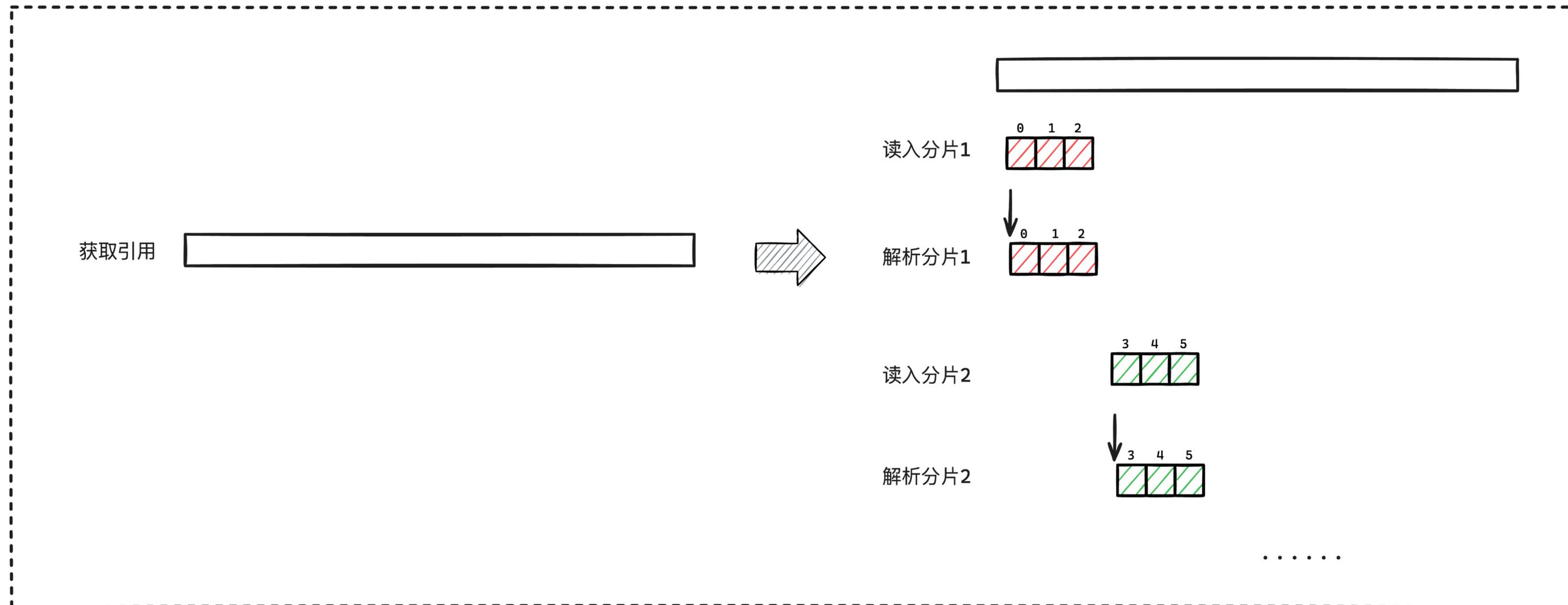


解析慢，效率低



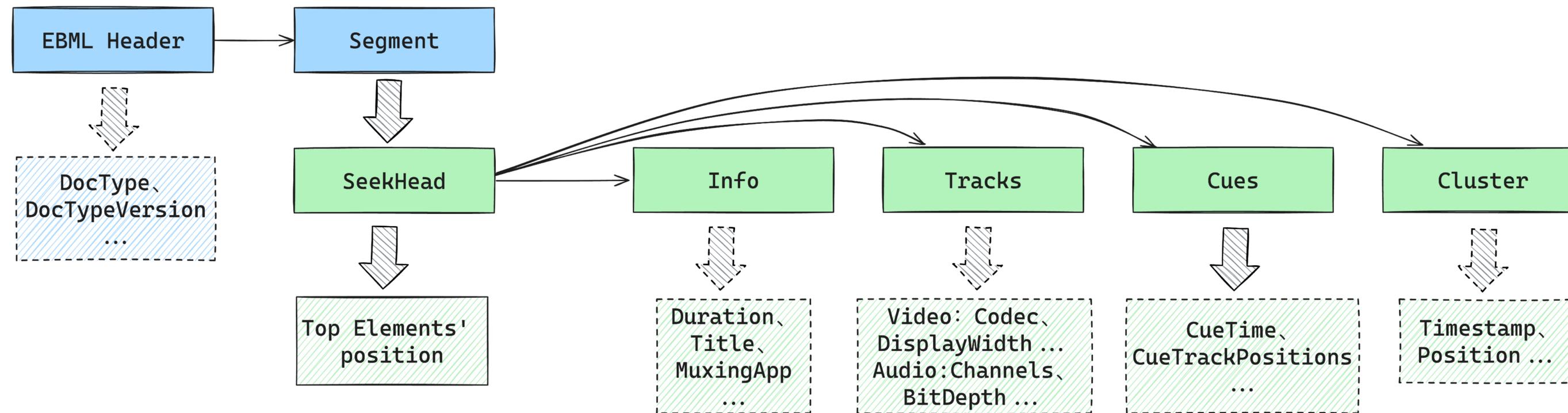
# 03. 方案 按需文件读取

- 读入：获取文件引用地址
- 使用：按需对文件进行切片，读取完成后销毁分片，切取下一分片



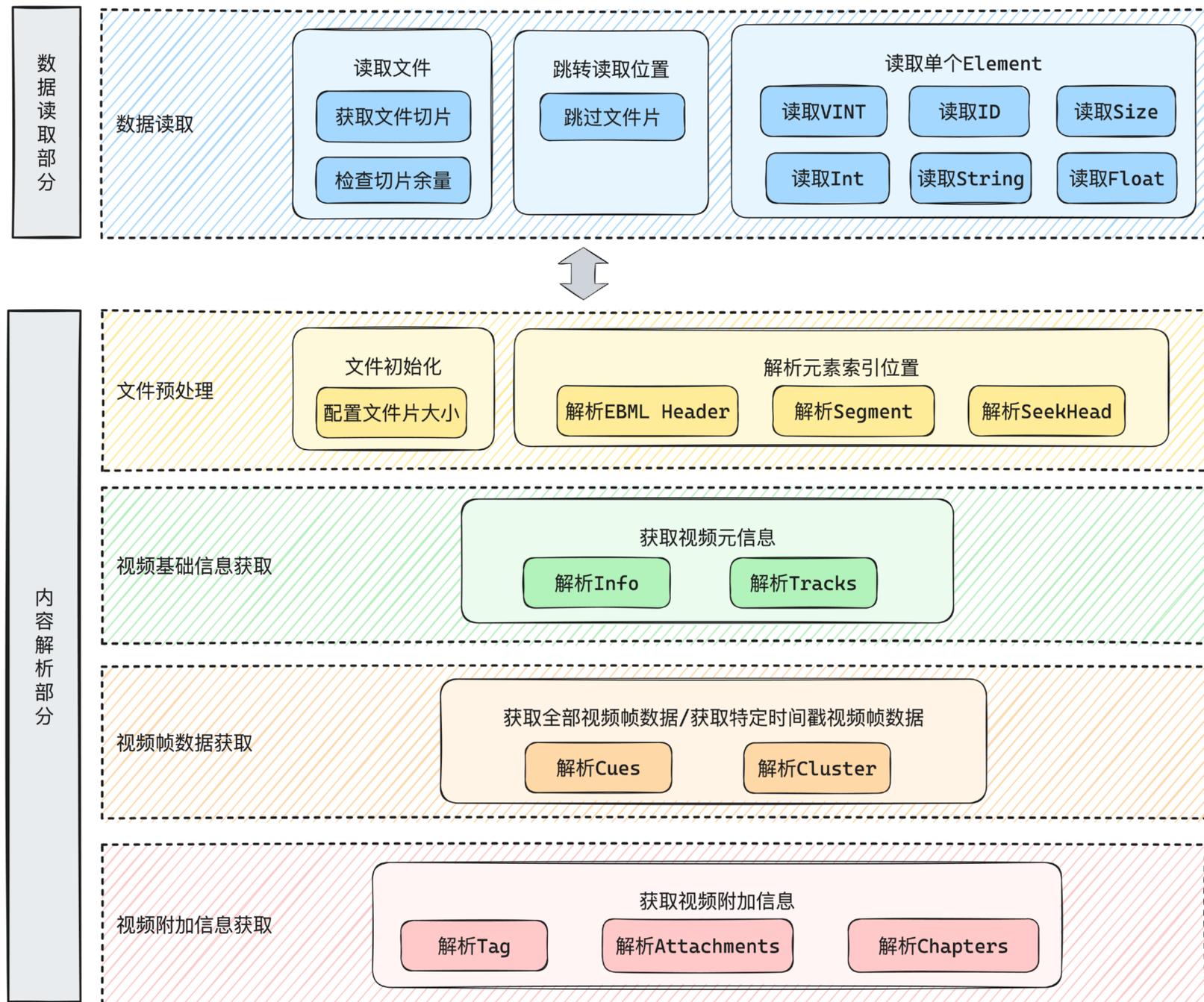
# 03. 方案 按需内容解析

- 先快速获取所有元素索引位置
- 按需解析



# 03. 方案

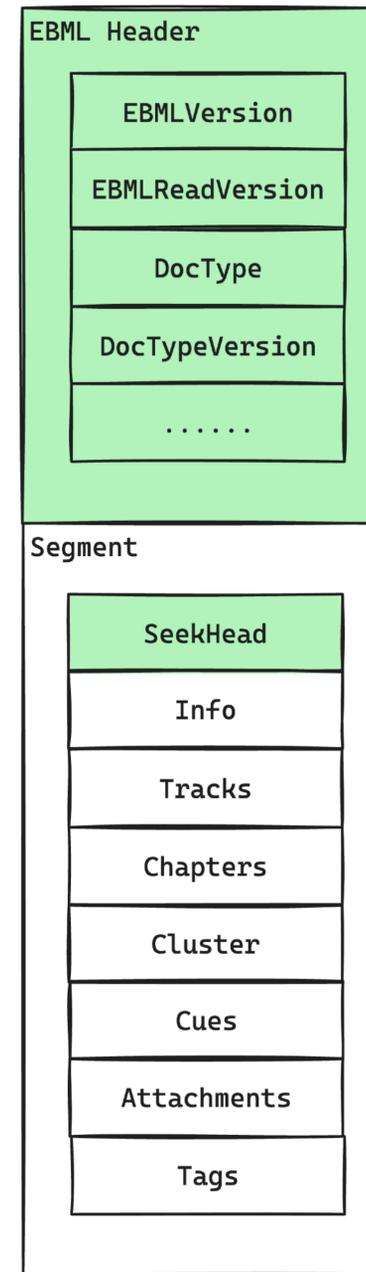
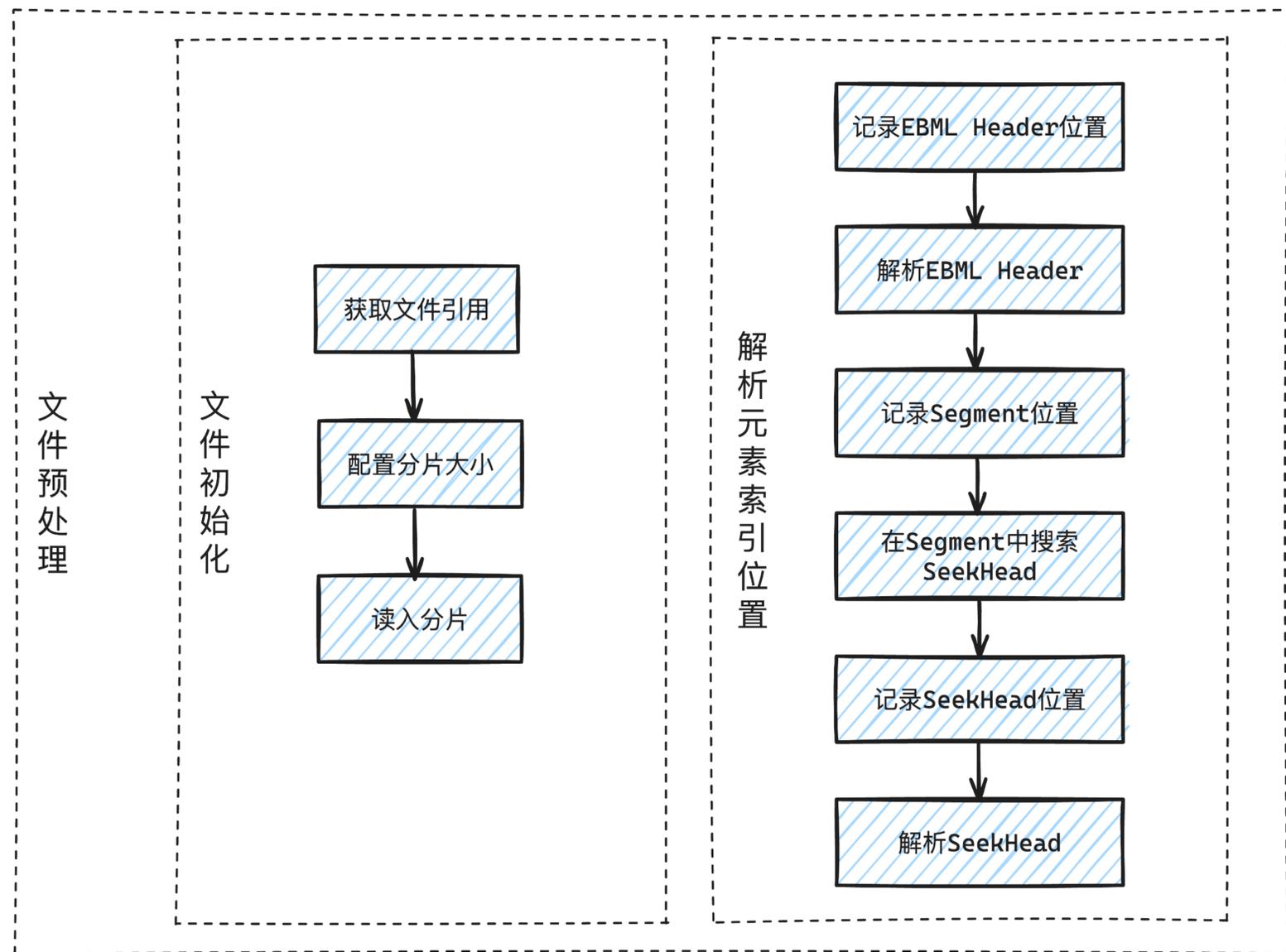
## 功能架构



# 03. 方案

## 文件预处理

```
const filePieceSize = 1 * 1024 * 1024  
await demuxer.initFile(file, filePieceSize)
```



## 元素位置map

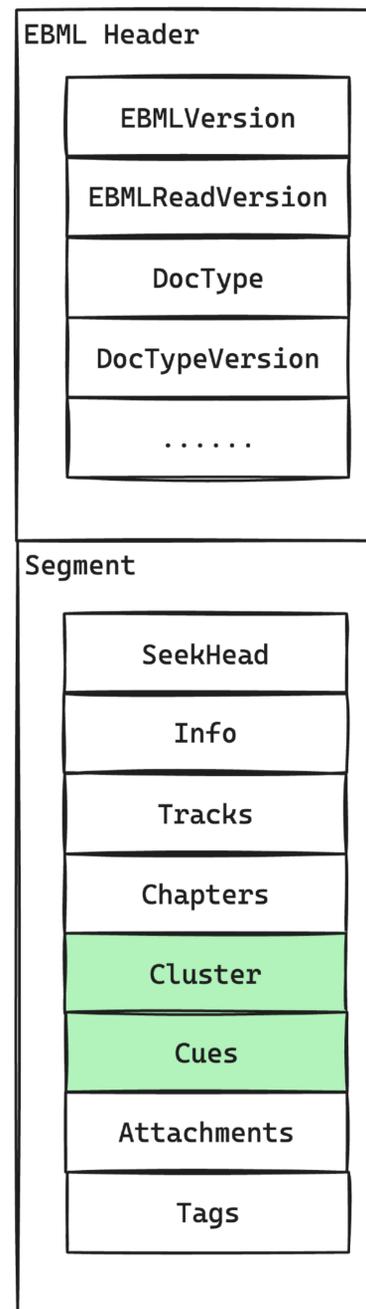
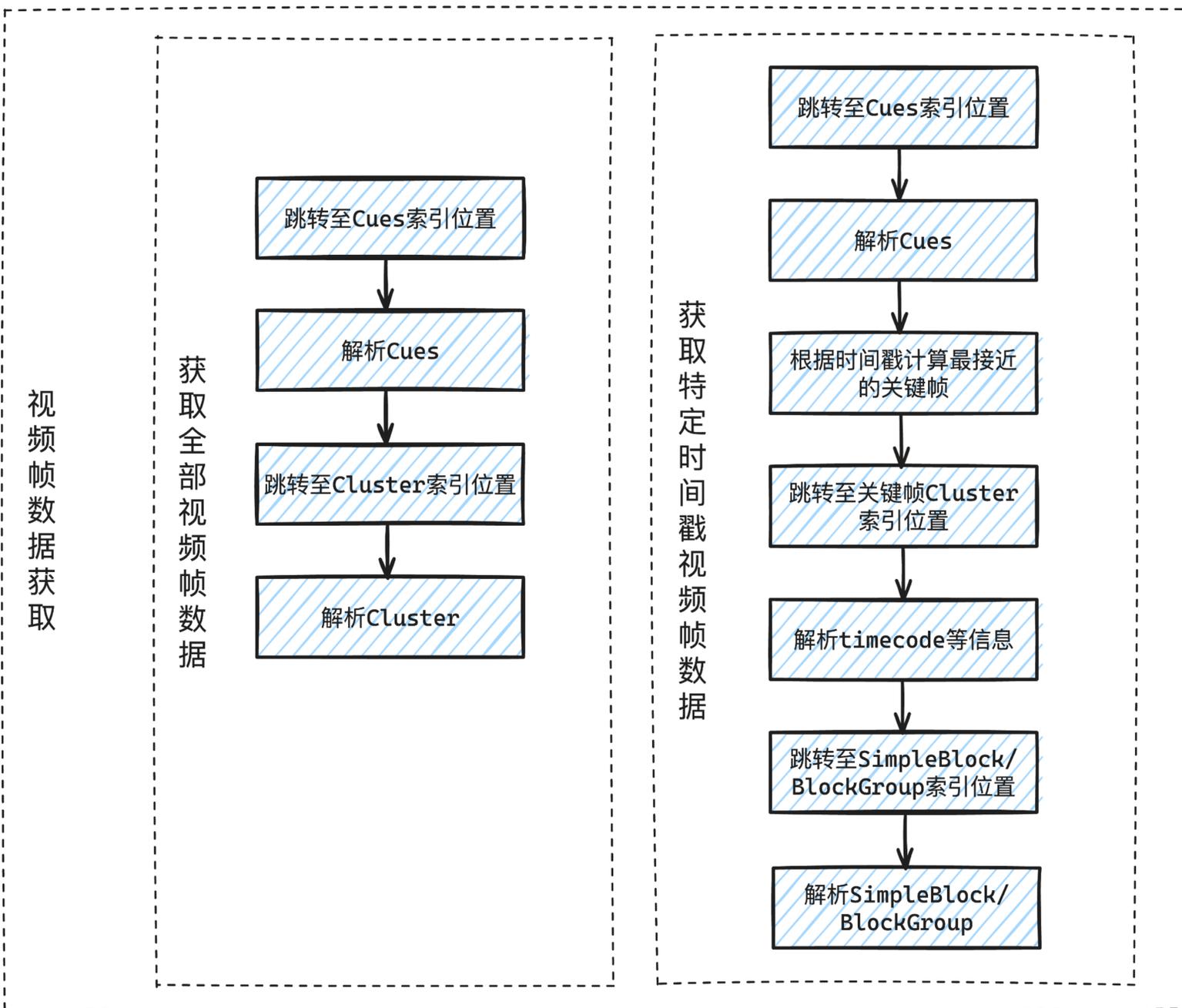
| Element Type | Position |
|--------------|----------|
| EBML Header  | 0        |
| Segment      | 40       |
| SeekHead     | 52       |
| Info         | 4151     |
| Tracks       | 4311     |
| Cues         | 31761776 |
| Cluster      | /        |
| Chapters     | /        |
| Attachments  | /        |
| Tags         | 31762539 |

# 03. 方案

## 视频帧数据获取

```
const data = await demuxer.getData()
```

```
const frame = await demuxer.seekFrame(10)
```



- data - The video data of the file
  - cues Array - The keyframes of the file
    - cueTime Number
    - cueTrackPositions
      - cueClusterPosition Number
      - cueRelativePosition Number
      - cueTrack Number
      - ...
    - ...
  - videoPackets Array
    - start Number
    - end Number
    - size Number
    - timestamp Number
    - isKeyframe Boolean
    - keyframeTimestamp Number
  - audioPackets Array
    - start Number
    - end Number
    - size Number
    - timestamp Number

## 03. 方案

mkv-demuxer: <https://www.npmjs.com/package/mkv-demuxer>

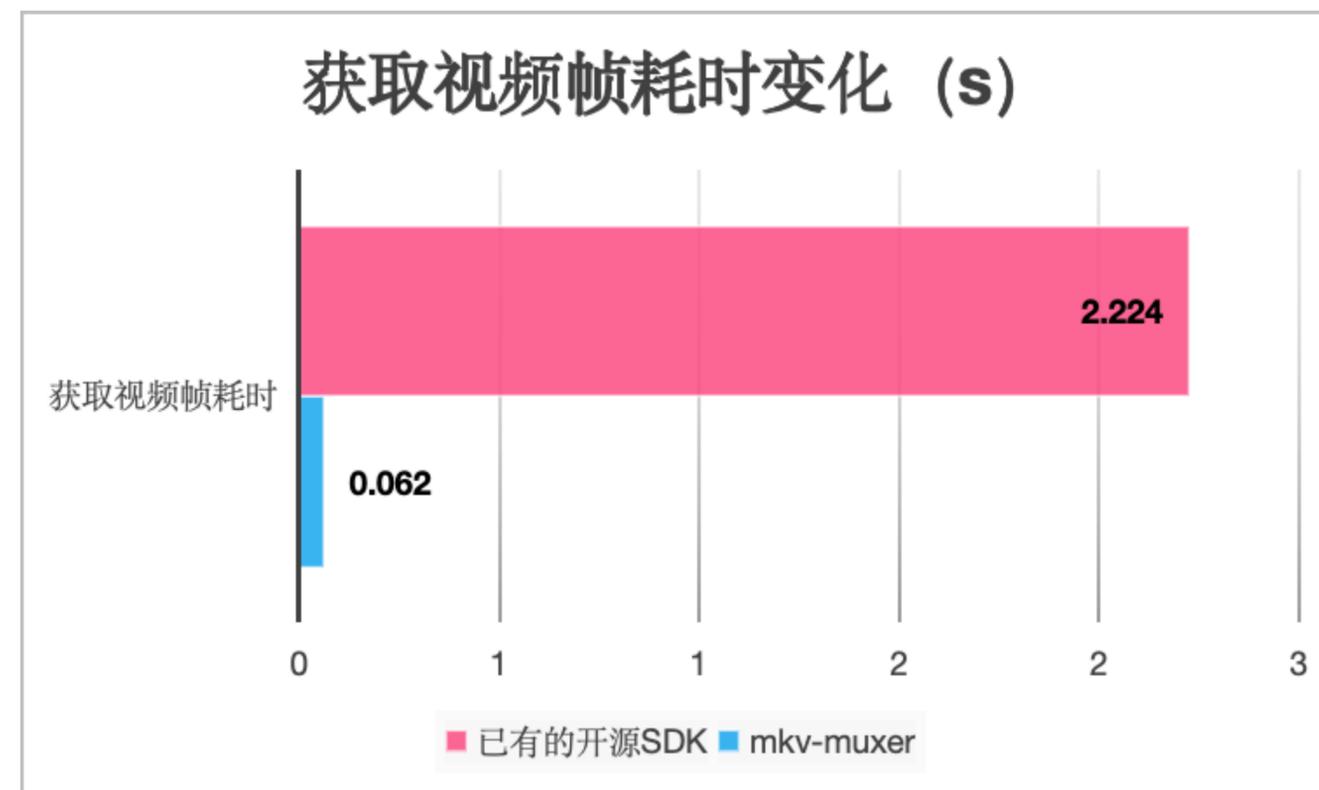
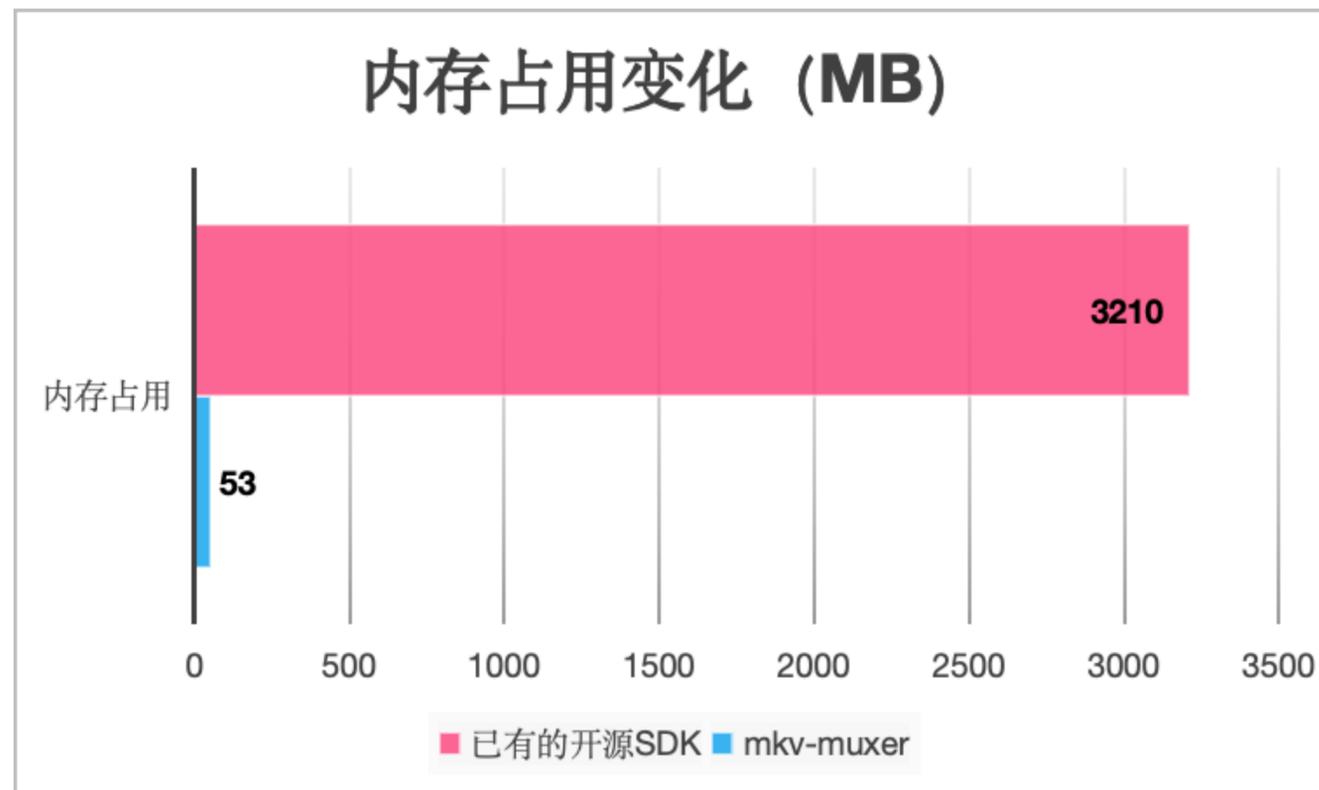
- 控制解析时的内存占用
- 使用高效快速的解析方式
- 提供了几个常用的API

```
import MkvDemuxer from 'mkv-demuxer'  
const demuxer = new MkvDemuxer()  
const fileSize = 1 * 1024 * 1024  
await demuxer.initFile(file, fileSize)  
const meta = await demuxer.getMeta()  
const data = await demuxer.getData()  
const frame = await demuxer.seekFrame(10)
```

# 03. 方案

以一个4K视频为例，其基本信息如下：

|      |             |
|------|-------------|
| 视频大小 | 1.61G       |
| 视频编码 | VP9         |
| 分辨率  | 3840 x 2160 |
| 码率   | 10023kb/s   |



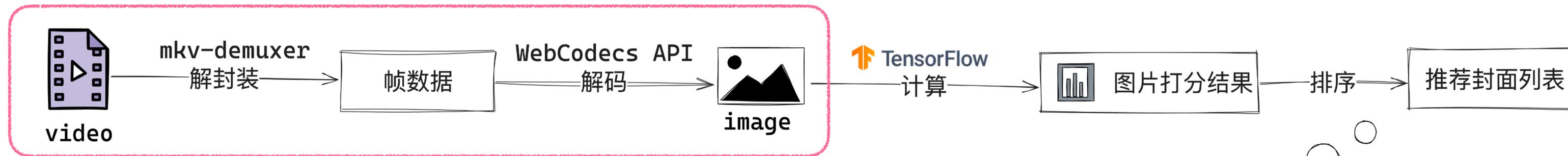
# 04.应用

04.Application

# 04.应用

## 推荐封面获取

### 升级方案

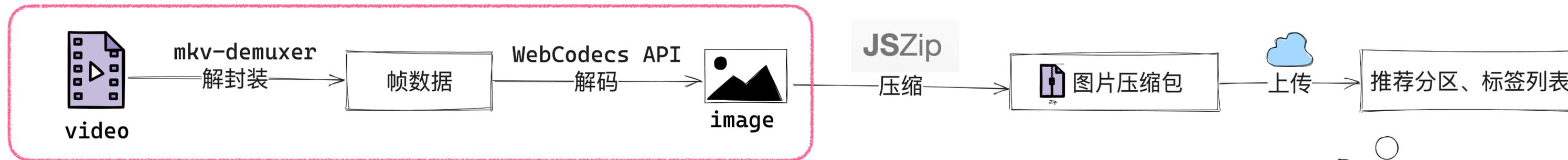


# 04.应用

推荐分区获取

推荐标签、话题获取

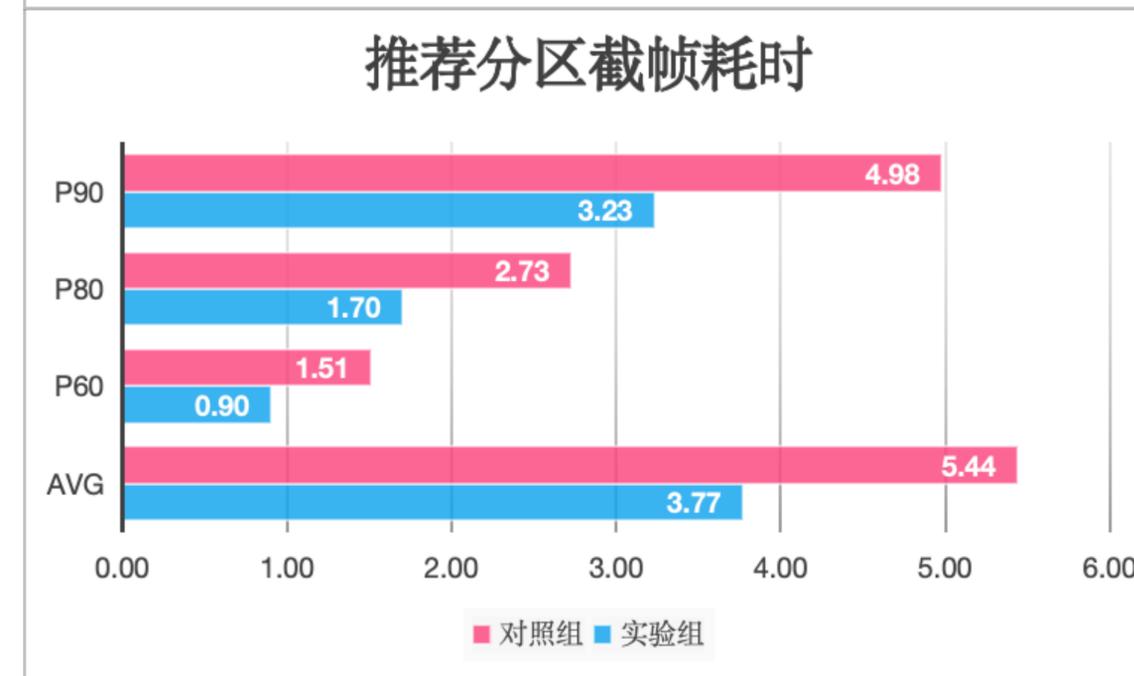
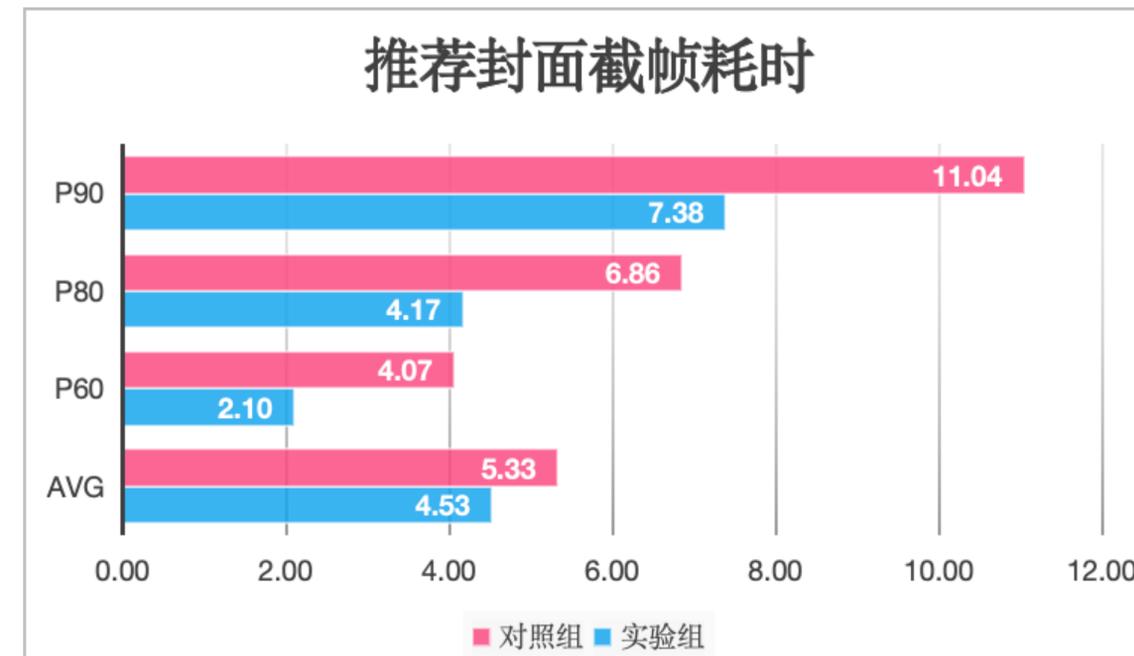
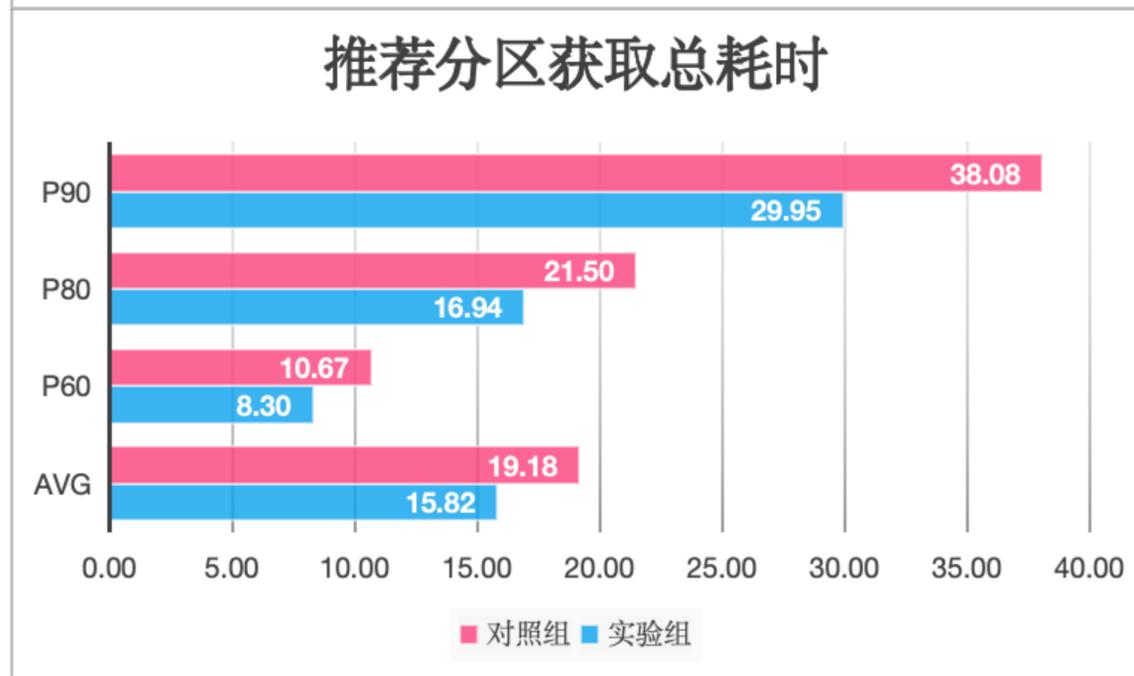
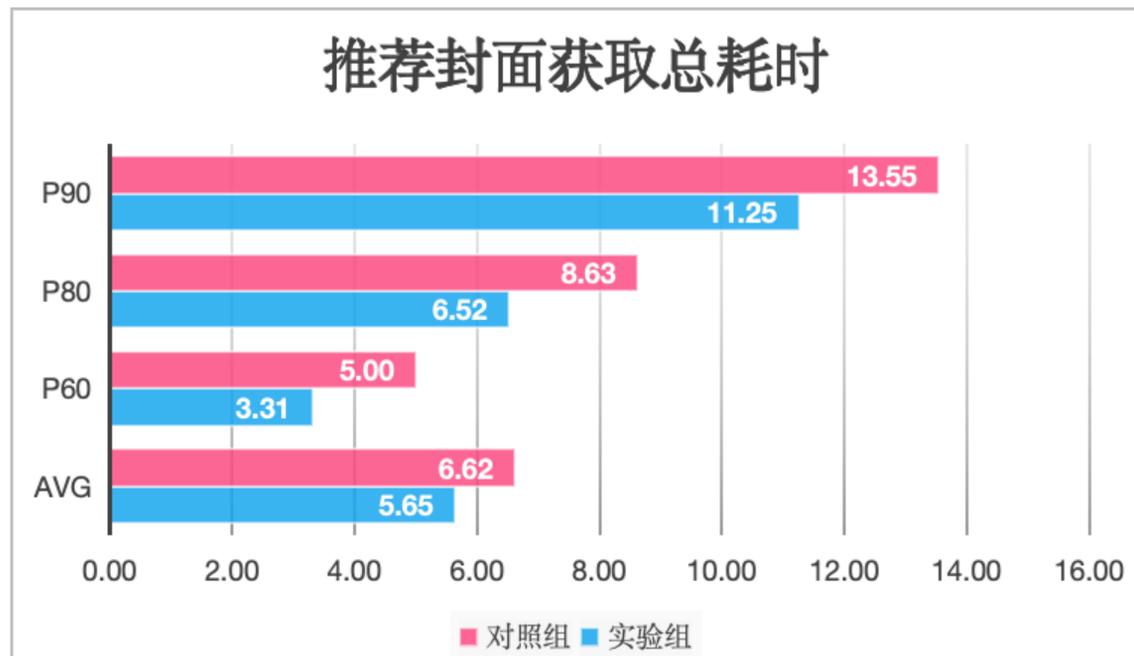
## 升级方案



# 04.应用

Matroska、WebM视频使用升级方案后:

- 推荐封面获取总耗时减少16.93%
  - 其中截帧环节耗时减少33.16%
- 推荐分区获取总耗时减少21.36%
  - 其中截帧环节耗时减少35.05%



# 05. 总结与展望

05. Conclusion and Outlook

## 05. 总结与展望

- 针对Web投稿页上的Matroska视频，可以利用mkv-demuxer的解封装能力，优化其它环节解析流程
- 进一步提升mkv-demuxer的解析能力，完善对Matroska格式的全面支持
- 跟随视频处理相关技术的发展持续进行探索

# 相关资料

EBML:

- [RFC 8794 – Extensible Binary Meta Language](#)
- [EBML: A binary encoding format](#)

Matroska:

- [Matroska Media Container Homepage](#)

WebM:

- <https://www.webmproject.org/>
- [WebM Format: Basic Facts, Compatibility, and WebM vs. MP4](#)

工具推荐:

- Hex Fiend
- MKVToolNix

项目地址及npm包:

- github仓库: [GitHub – SuperYanjun/mkv-demuxer](#)
- npm包: <https://www.npmjs.com/package/mkv-demuxer>



谢谢