Infrastructure

Ваш плеер работает неправильно

Константин Петряев
Yandex Infrastructure

Константин Петряев

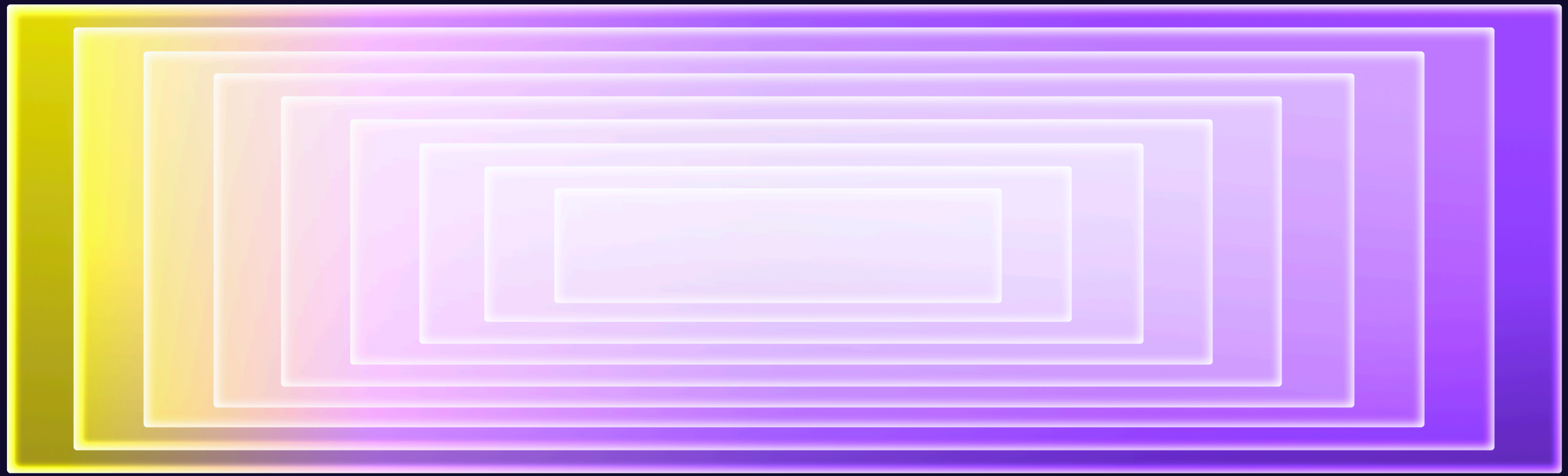
9 лет занимаюсь web-разработкой

5 лет разрабатываю плеер

Написал плеер «с нуля»

План доклада

1. Проблема с качеством
2. Как считать скорость сети
3. hls.js
4. shaka-player
5. Лекарство



Проблема с качеством

Preserve log | Disable cache | 1700kbp/s

fragment | Invert | Hide data URLs | Hide extension URLs

All | Fetch/XHR | JS | CSS | Img | Media | Font | Doc | WS | Wasm | Manifest | Other

Blocked response cookies | Blocked requests | 3rd-party requests

5000 ms | 10000 ms | 15000 ms | 20000 ms | 25000 ms | 30000 ms | 35000 ms | 40000 ms

Name	Size	Time	Waterfall
fragment-3-vid1.m4s?id=0.yud11d3qi…	172 kB	1.62 s	
fragment-4-aid5.m4s?id=0.yud11d3qi…	226 kB	1.87 s	
fragment-4-vid1.m4s?id=0.yud11d3qi…	170 kB	1.61 s	
fragment-5-vid1.m4s?id=0.yud11d3qi…	102 kB	876 ms	
fragment-5-aid5.m4s?id=0.yud11d3qi…	226 kB	1.47 s	
fragment-6-vid1.m4s?id=0.yud11d3qi…	138 kB	1.26 s	
fragment-6-aid5.m4s?id=0.yud11d3qi…	226 kB	1.68 s	
fragment-7-vid1.m4s?id=0.yud11d3qi…	178 kB	1.65 s	
fragment-7-aid5.m4s?id=0.yud11d3qi…	226 kB	1.88 s	
fragment-8-vid1.m4s?id=0.yud11d3qi…	155 kB	1.47 s	
fragment-8-aid5.m4s?id=0.yud11d3qi…	226 kB	1.80 s	
fragment-9-aid5.m4s?id=0.yud11d3qi…	226 kB	1.80 s	
fragment-9-vid1.m4s?id=0.yud11d3qi…	154 kB	1.46 s	
fragment-10-aid5.m4s?id=0.yud11d3q…	226 kB	1.76 s	
fragment-10-vid1.m4s?id=0.yud11d3q…	147 kB	1.39 s	
fragment-11-aid5.m4s?id=0.yud11d3q…	4.1 kB	625 ms	
fragment-11-vid1.m4s?id=0.yud11d3q…	4.1 kB	663 ms	

22 / 69 requests | 3.8 MB / 5.4 MB transferred | 3.9 MB / 5.6 MB resources | Finish: 36.57 s

Предыстория

Поток с большими аудио сегментами

Скорость сети в 1.7 Мб/с

Сети хватает на 360p + Аудио

Плеер качает 144p + Аудио

Сегмент	Битрейт
Аудио	452 Кбит/с
144p	412 Кбит/с
240p	812 Кбит/с
360p	947 Кбит/с
478p	1 615 Кбит/с

Кто отвечает за выбор качества в плеере?

Adaptive BitRate Manager (ABR)

Выбирает видео/аудио дорожку

На вход получает список

На выходе — одна дорожка или сочетание

Ориентируется на сетевые параметры*

https://clck.ru/36BCAk

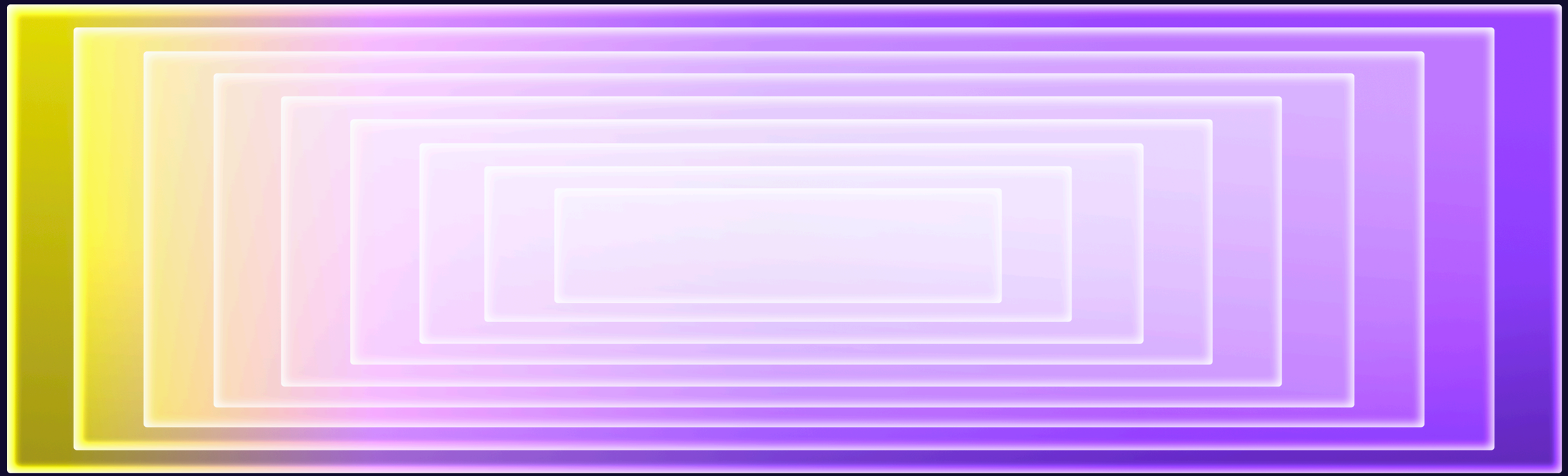
Эстиматор

Одна из важнейших частей ABR

Собирает данные о сети

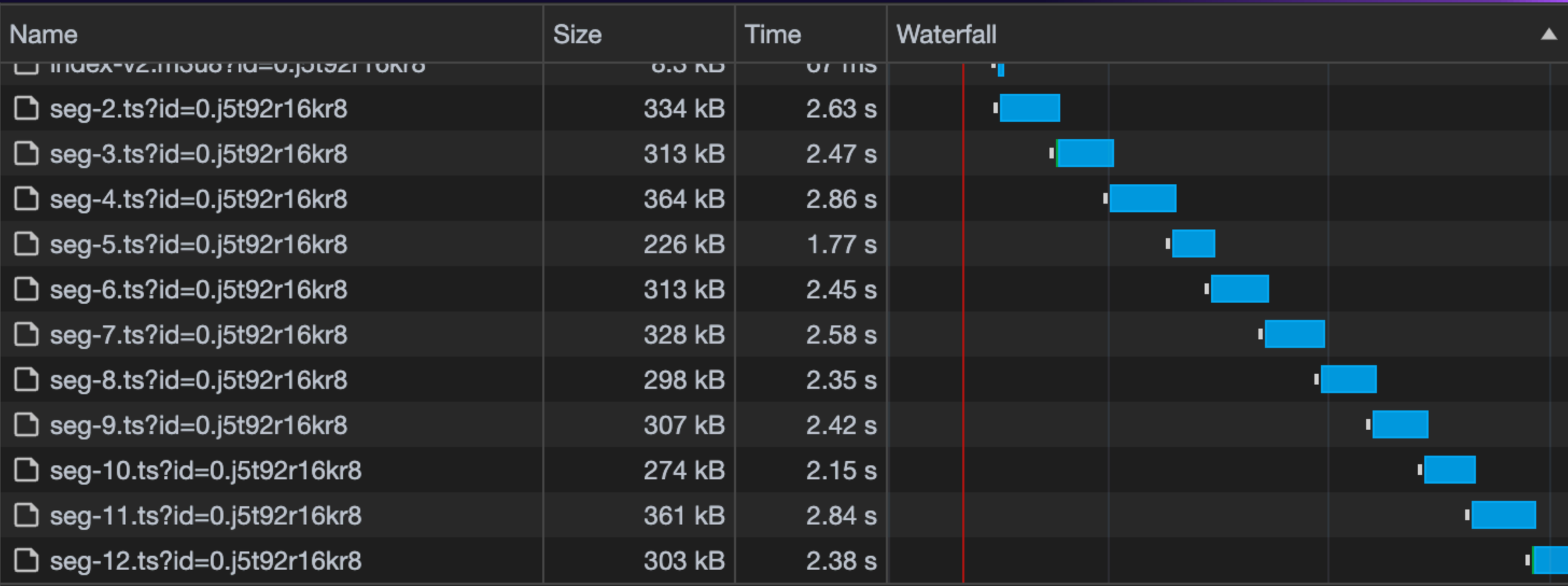
Перерабатывает в удобный вид

Источник правды о сети для плеера



Поговорим о базе

Как посчитать скорость сети



Name	Size	Time	Waterfall
index-v2.m3u8?id=0.j5t92r16kr8	8.3 kB	67 ms	
seg-2.ts?id=0.j5t92r16kr8	334 kB	2.63 s	
seg-3.ts?id=0.j5t92r16kr8	313 kB	2.47 s	
seg-4.ts?id=0.j5t92r16kr8	364 kB	2.86 s	
seg-5.ts?id=0.j5t92r16kr8	226 kB	1.77 s	
seg-6.ts?id=0.j5t92r16kr8	313 kB	2.45 s	
seg-7.ts?id=0.j5t92r16kr8	328 kB	2.58 s	
seg-8.ts?id=0.j5t92r16kr8	298 kB	2.35 s	
seg-9.ts?id=0.j5t92r16kr8	307 kB	2.42 s	
seg-10.ts?id=0.j5t92r16kr8	274 kB	2.15 s	
seg-11.ts?id=0.j5t92r16kr8	361 kB	2.84 s	
seg-12.ts?id=0.j5t92r16kr8	303 kB	2.38 s	

Это элементарно!

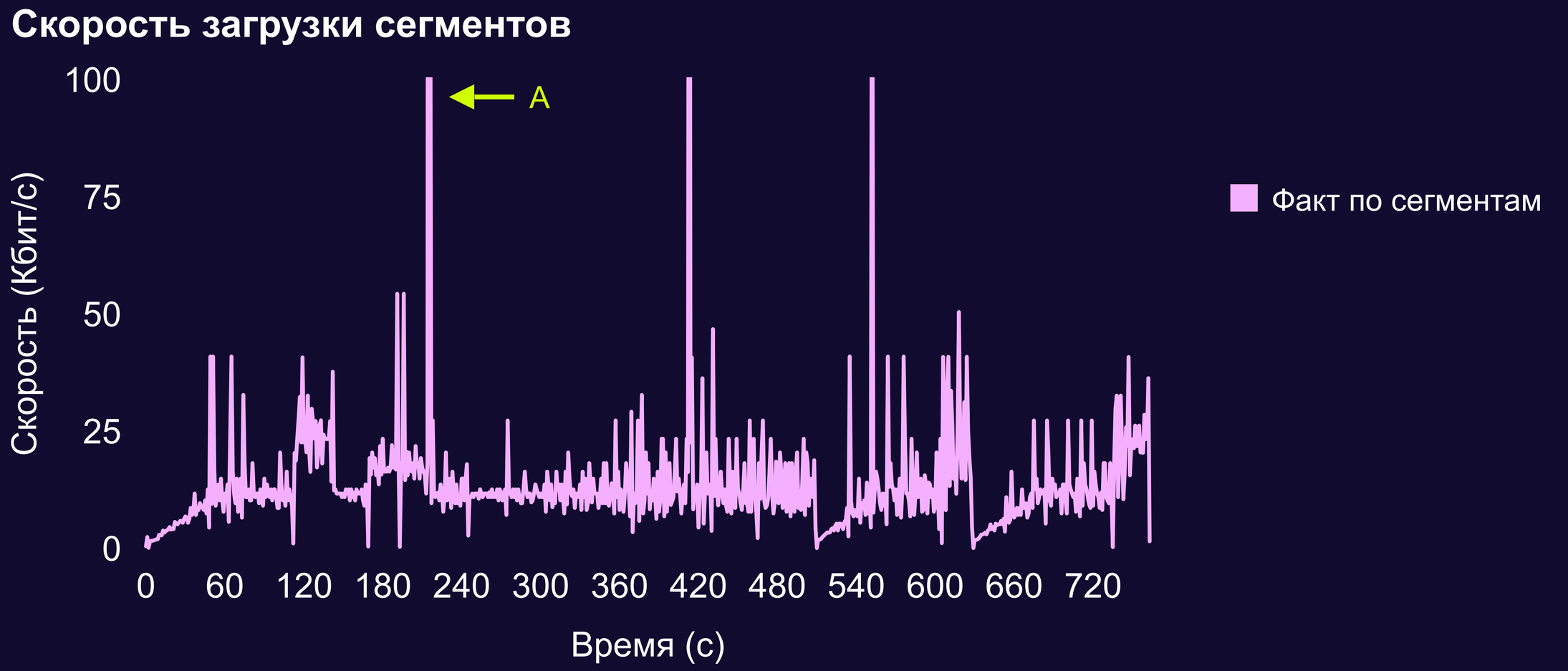
$$Bandwidth = Size/Time$$

Так делает hls.js

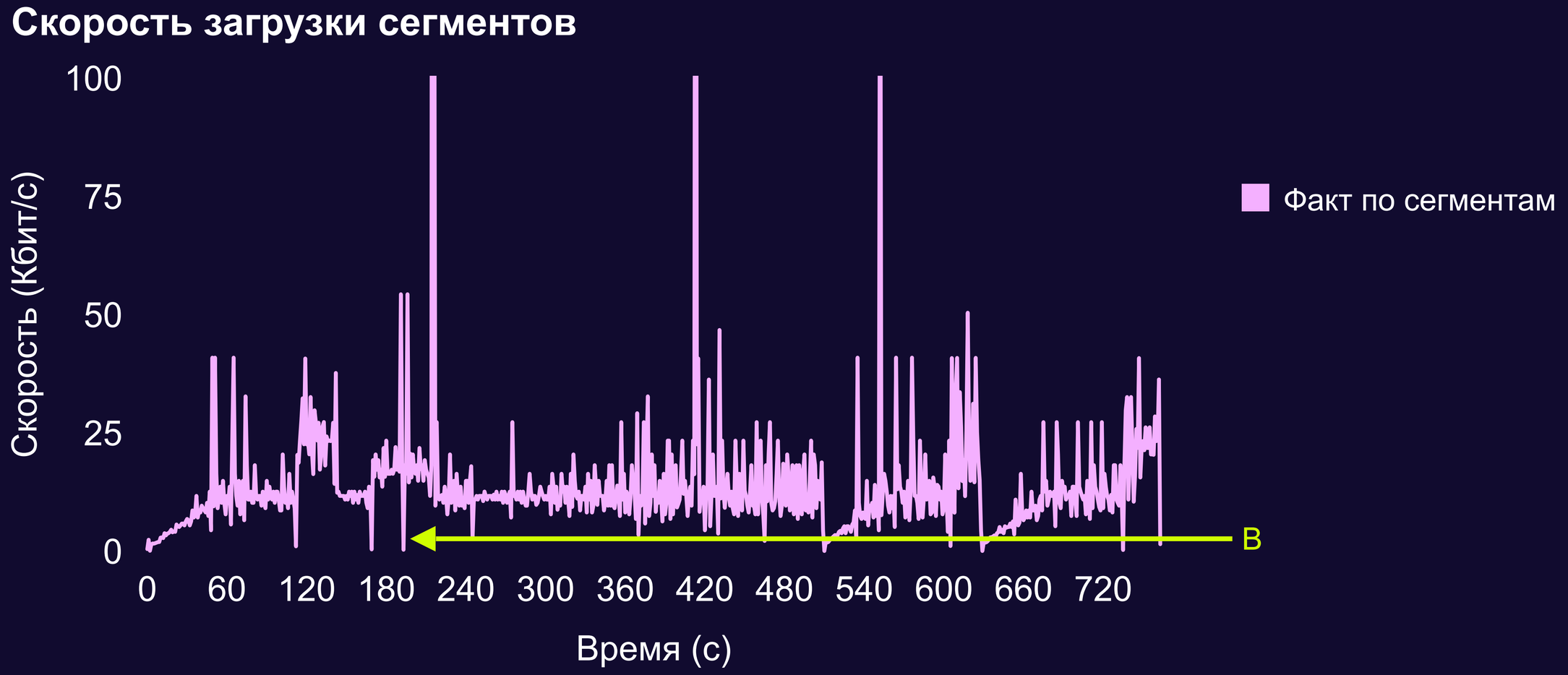
```
function onFragLoaded(fragmentData) {
  const {
    loading,
    size /* Bits */
  } = fragmentData;
  const loadingDuration /* Seconds */ = loading.end - loading.start;

  bwEstimator.sample(loadingDuration, size);
}
```

Скорость — величина переменная

Скорость — величина переменная



Скорость — величина переменная



Данные надо сглаживать

Для этого почти всегда используется EWMA

Exponentially Weighted Moving Average

Экспоненциальная взвешенная скользящая средняя

$$EWMA_t = \alpha p_t + (1 - \alpha) EWMA_{t-1}$$

p_t – значение измерения в конкретный момент времени t

α – сглаживающая константа от 0 до 1

EWMA — наглядно

Популярные библиотеки

hls.js

utils/ewma-bandwidth-estimator.ts

dash.js

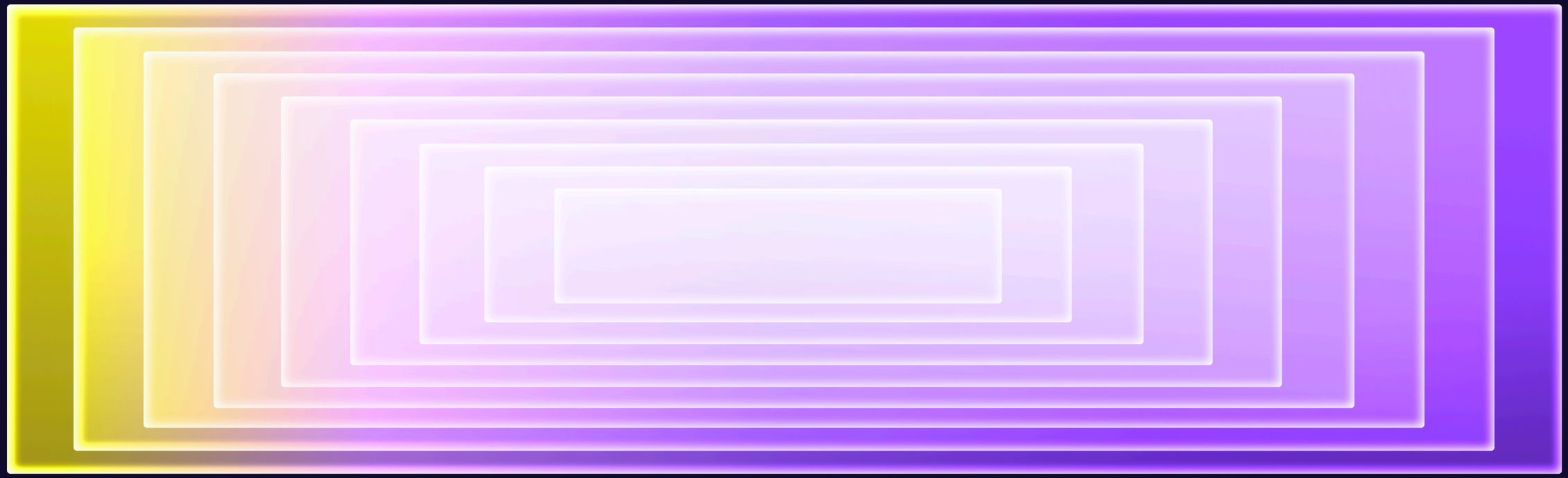
streaming/rules/
ThroughputHistory.js

shaka-player

abr/ewma_bandwidth_estimator.js

rx-player

core/adaptive/utils/
bandwidth_estimator.ts



Как себя ведёт hls.js?

В devtools кривой network throttling

Проверяем цифры

Фактически качаем

$$7.6 * 8 * 2^{10} / 59.72 = 1042.5 \text{ Кбит/с}$$

Эстимейт

$$844902 / 1024 = 825 \text{ Кбит/с}$$

Разница в 21%

hls.js ошибается в оценке пропускной способности сети

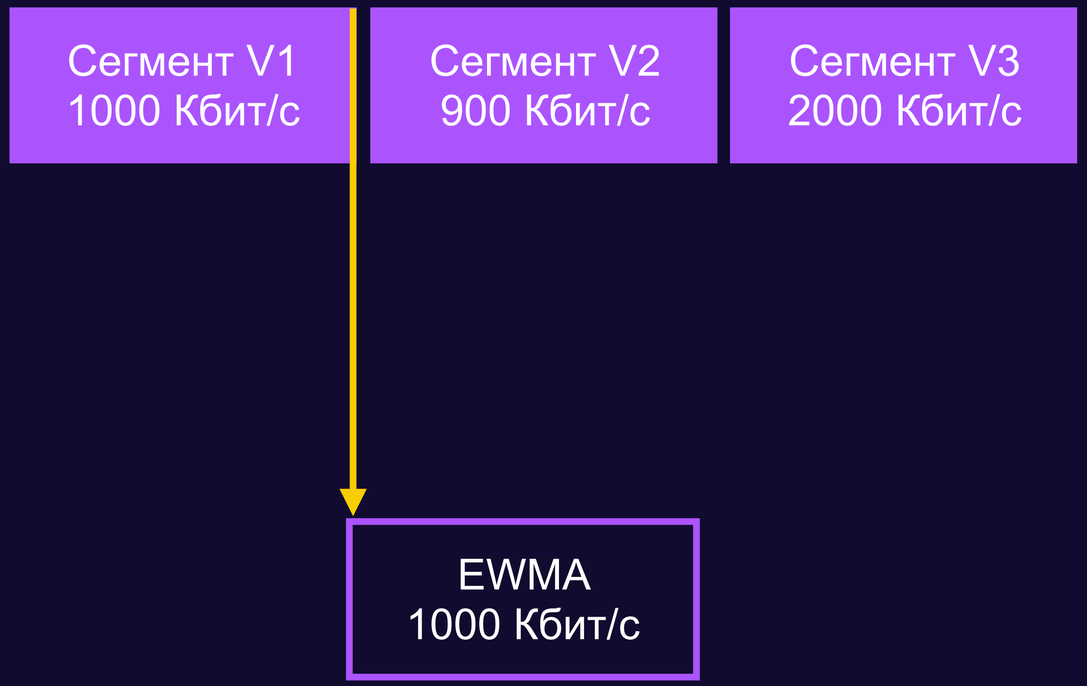
Попробуем провести процесс подсчётов «руками»

$$EWMA_t = \alpha p_t + (1 - \alpha) EWMA_{t-1} \qquad \alpha = 0.1$$

Сегмент V1 1000 Кбит/с	Сегмент V2 900 Кбит/с	Сегмент V3 2000 Кбит/с

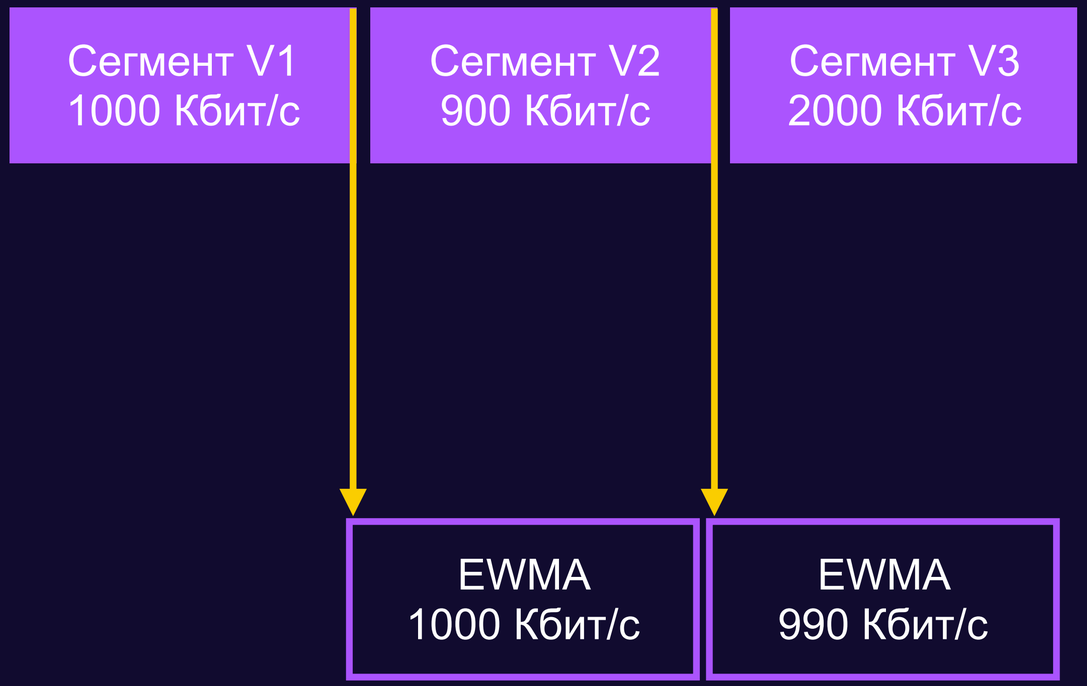
Попробуем провести процесс подсчётов «руками»

$$EWMA_t = \alpha p_t + (1 - \alpha) EWMA_{t-1} \qquad \alpha = 0.1$$



Попробуем провести процесс подсчётов «руками»

$$EWMA_t = \alpha p_t + (1 - \alpha) EWMA_{t-1} \qquad \alpha = 0.1$$



Попробуем провести процесс подсчётов «руками»

$$EWMA_t = \alpha p_t + (1 - \alpha) EWMA_{t-1} \qquad \alpha = 0.1$$

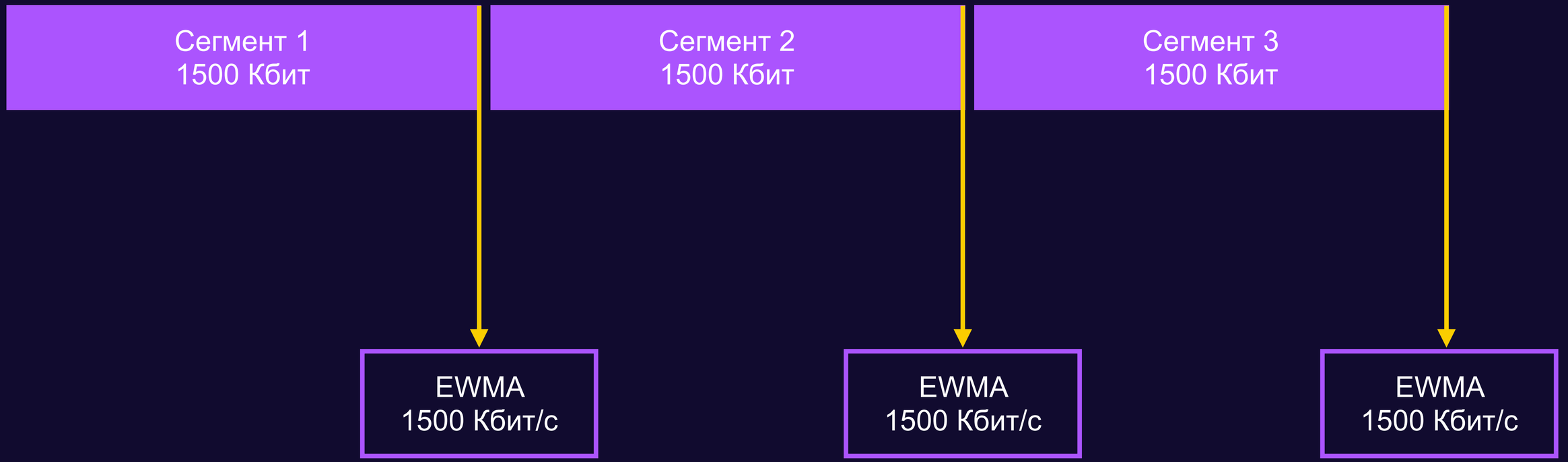
Упростим подсчёт

$$EWMA_t = \alpha p_t + (1 - \alpha) EWMA_{t-1} \qquad \alpha = 0.1$$

Сегмент 1 1500 Кбит	Сегмент 2 1500 Кбит	Сегмент 3 1500 Кбит

Упростим подсчёт

$$EWMA_t = \alpha p_t + (1 - \alpha) EWMA_{t-1} \qquad \alpha = 0.1$$

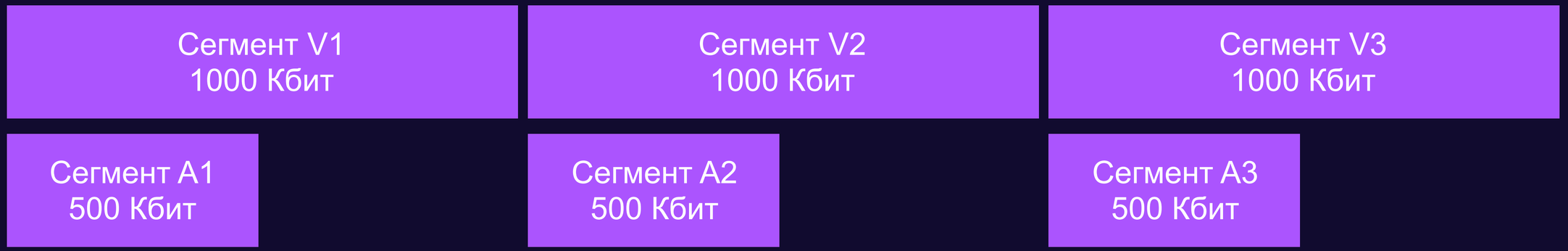


В HLS v3 модель работает

В HLS v7+ дорожки можно разделять

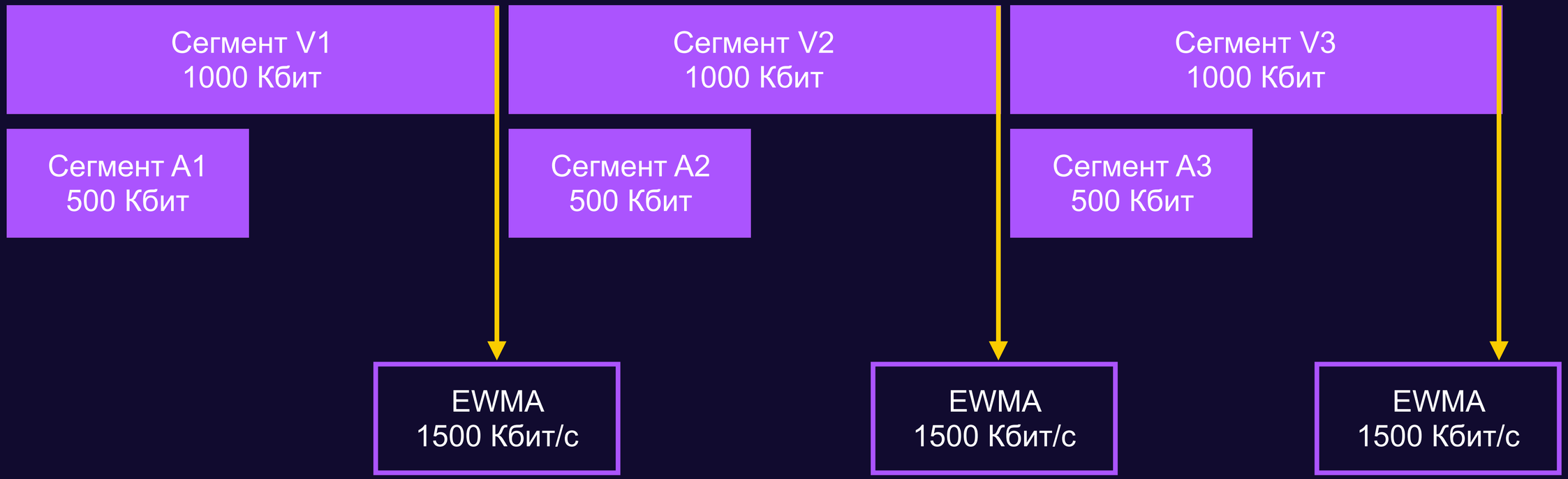
Их разделяют и грузят так

$$EWMA_t = \alpha p_t + (1 - \alpha) EWMA_{t-1} \qquad \alpha = 0.1$$



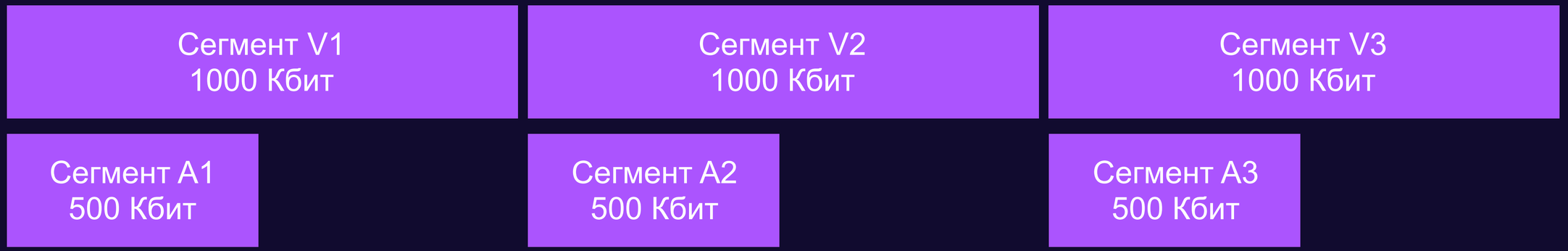
Их разделяют и грузят так

$$EWMA_t = \alpha p_t + (1 - \alpha) EWMA_{t-1} \qquad \alpha = 0.1$$



А на самом деле

$$EWMA_t = \alpha p_t + (1 - \alpha) EWMA_{t-1} \qquad \alpha = 0.1$$



А на самом деле

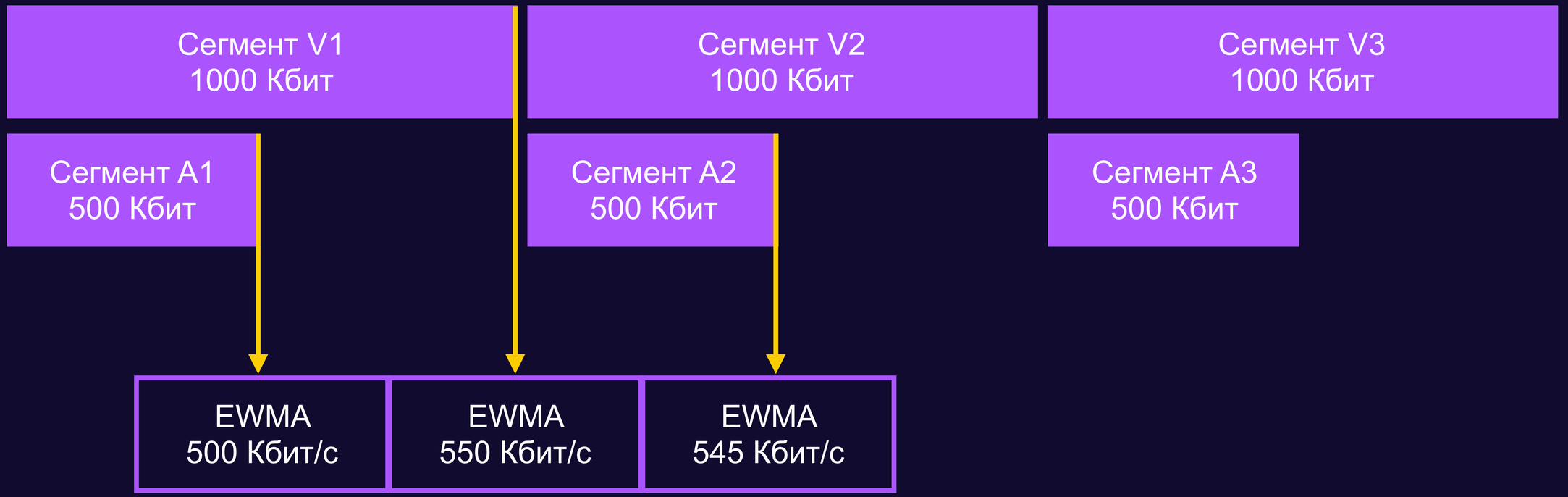
$$EWMA_t = \alpha p_t + (1 - \alpha) EWMA_{t-1} \qquad \alpha = 0.1$$

А на самом деле

$$EWMA_t = \alpha p_t + (1 - \alpha) EWMA_{t-1} \qquad \alpha = 0.1$$

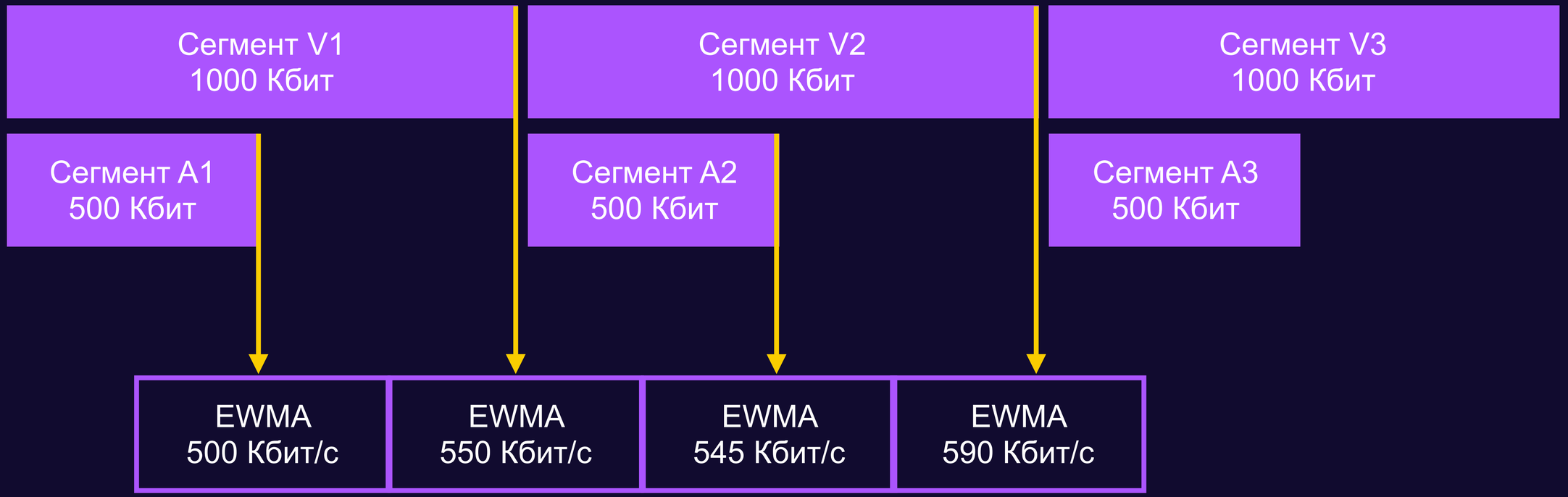
А на самом деле

$$EWMA_t = \alpha p_t + (1 - \alpha) EWMA_{t-1} \qquad \alpha = 0.1$$



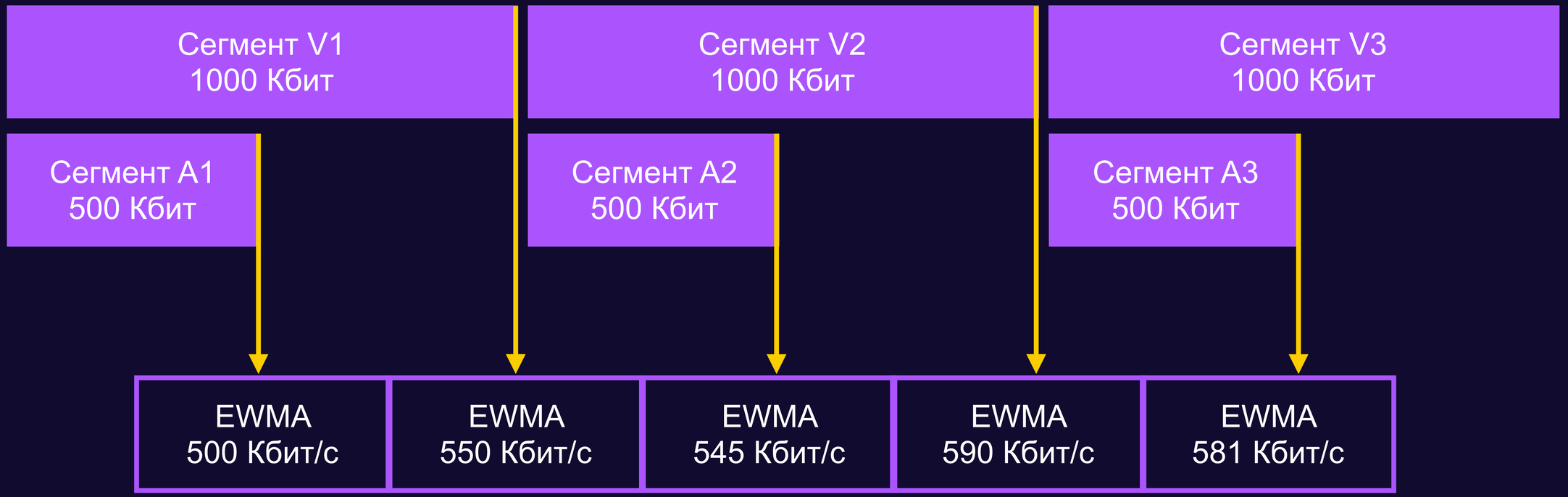
А на самом деле

$$EWMA_t = \alpha p_t + (1 - \alpha) EWMA_{t-1} \qquad \alpha = 0.1$$



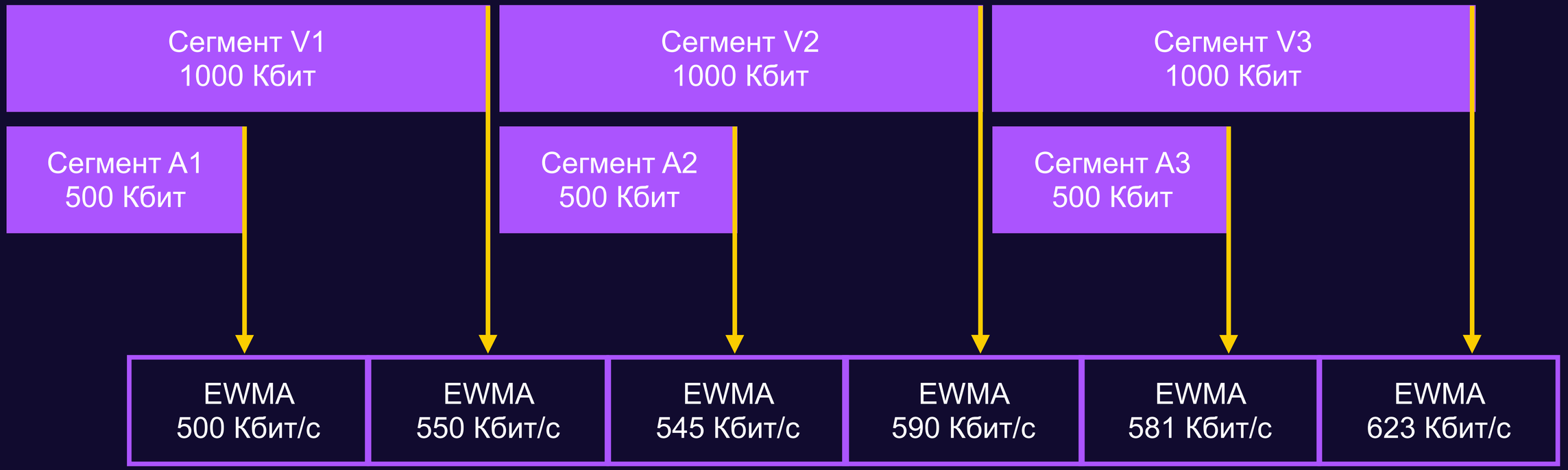
А на самом деле

$$EWMA_t = \alpha p_t + (1 - \alpha) EWMA_{t-1} \qquad \alpha = 0.1$$



А на самом деле

$$EWMA_t = \alpha p_t + (1 - \alpha) EWMA_{t-1} \qquad \alpha = 0.1$$



Всё не так

$$EWMA_t = \alpha p_t + (1 - \alpha) EWMA_{t-1}$$

$$EWMA_{t_v} = \alpha p_{t_v} + (1 - \alpha) EWMA_{t_a-1}$$

$$EWMA_{t_a-1} = \alpha p_{t_a-1} + (1 - \alpha) EWMA_{t_v-2}$$

Модель некорректно работает с двумя потоками загружаемых данных

hls.js пытается это обходить

Посмотрим на код еще раз

```
function onFragLoaded(fragmentData) {
  const {
    loading,
    size /* Bits */
  } = fragmentData;
  const loadingDuration /* Seconds */ = loading.end - loading.start;

  bwEstimator.sample(loadingDuration, size);
}
```

Есть нюанс

```
function onFragLoaded(fragmentData) {
  if (ignoreFragment(fragmentData)) return;
  const {
    loading,
    size /* Bits */
  } = fragmentData;
  const loadingDuration /* Seconds */ = loading.end - loading.start;

  bwEstimator.sample(loadingDuration, size);
}
```

Вот он

```
function ignoreFragment(fragmentData) {
  // Only count non-alt-audio frags
  return frag.type !== PlaylistLevelType.MAIN ||
    frag.sn === ‘initSegment’
}
```

hls.js игнорирует аудио-дорожку в контексте сети

В первом приближении

Это нормальное инженерное решение

- Нужно качать аудио
- Влиять на него нельзя

Надо разобраться

Поставим себя на место hls.js

Поставим себя на место hls.js

Поставим себя на место hls.js

2000 Кбит/с


Сегмент V1
1000 Кбит
Сегмент A1
500 Кбит
2000 Кбит/с
500 + 500 Кбит
0.5 с
2000 Кбит/с
500 Кбит
0.25 с

Получаем меньше

Получаем меньше

Получаем меньше

1500 Кб/с

Можем качать

1333 Кб/с

Насчитали

1400 Кб/с

Следующее качество

Нам нужны данные о битрейте аудио

Как их получить?

EXT-X-BITRATE в level-плейлистах

Поддержка запланирована в hls.js v1.8

Сейчас hls.js v1.4

Как их получить?

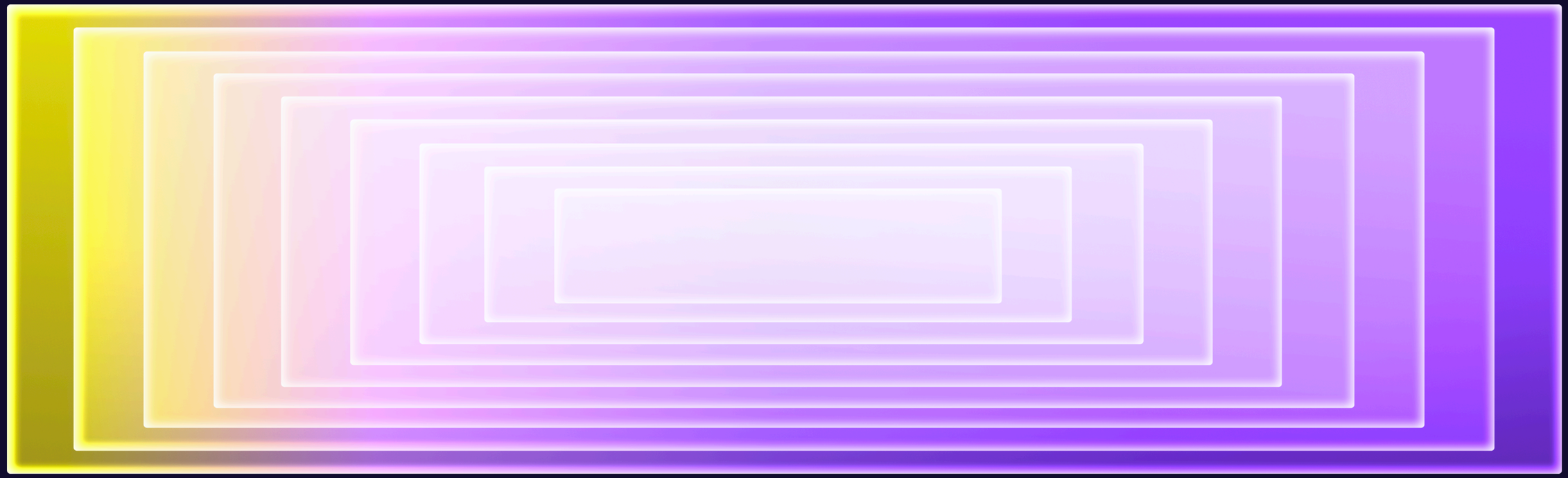
EXT-X-BITRATE в level-плейлистах

Поддержка запланирована в hls.js v1.8

Сейчас hls.js v1.4

HLS как стандарт
плох для медленных сетей

Посмотрим на DASH



Что с shaka-player?

Лимит на размер сэмпла

```
class BandwidthEstimator {
  minBites = 16e3;
  sample(durationMs, numBytes) {
    if (numBytes < this.minBytes) {
      return;
    }
    const bandwidth = 8000 * numBytes / durationMs;
    this.realSample(bandwidth);
  }
}
```

Что видим?

1

shaka-player грузит аудио и видео параллельно

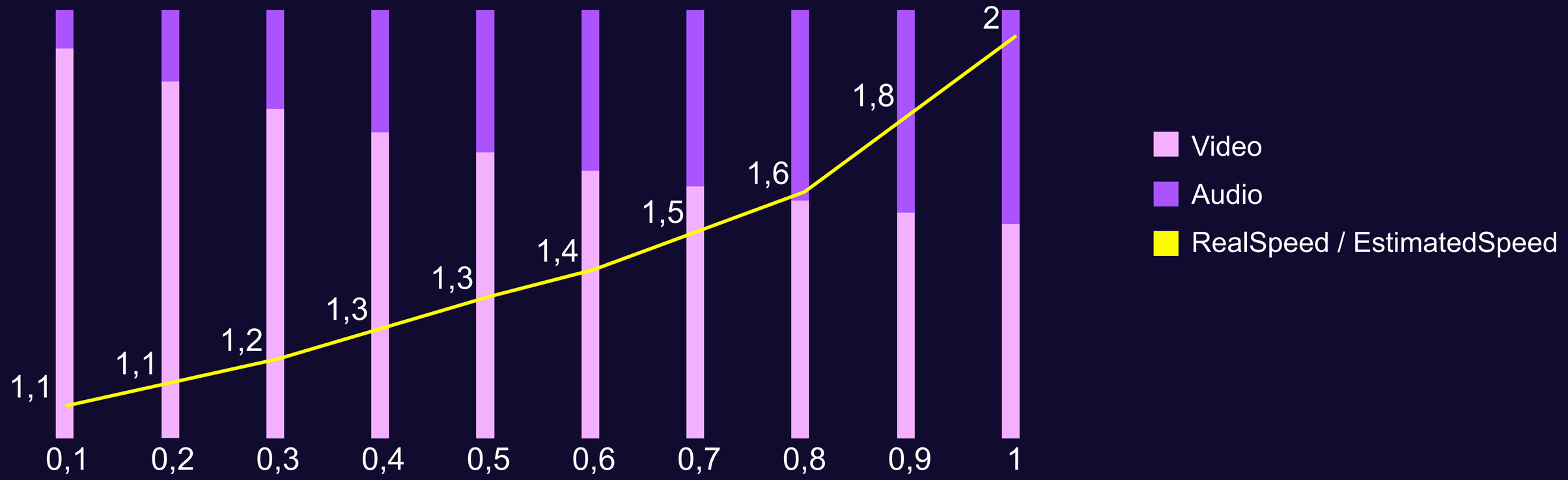
2

Учитывает сэмплы и аудио, и видео

3

Ошибается в оценке битрейта

Плеера недооценивают скорость сети



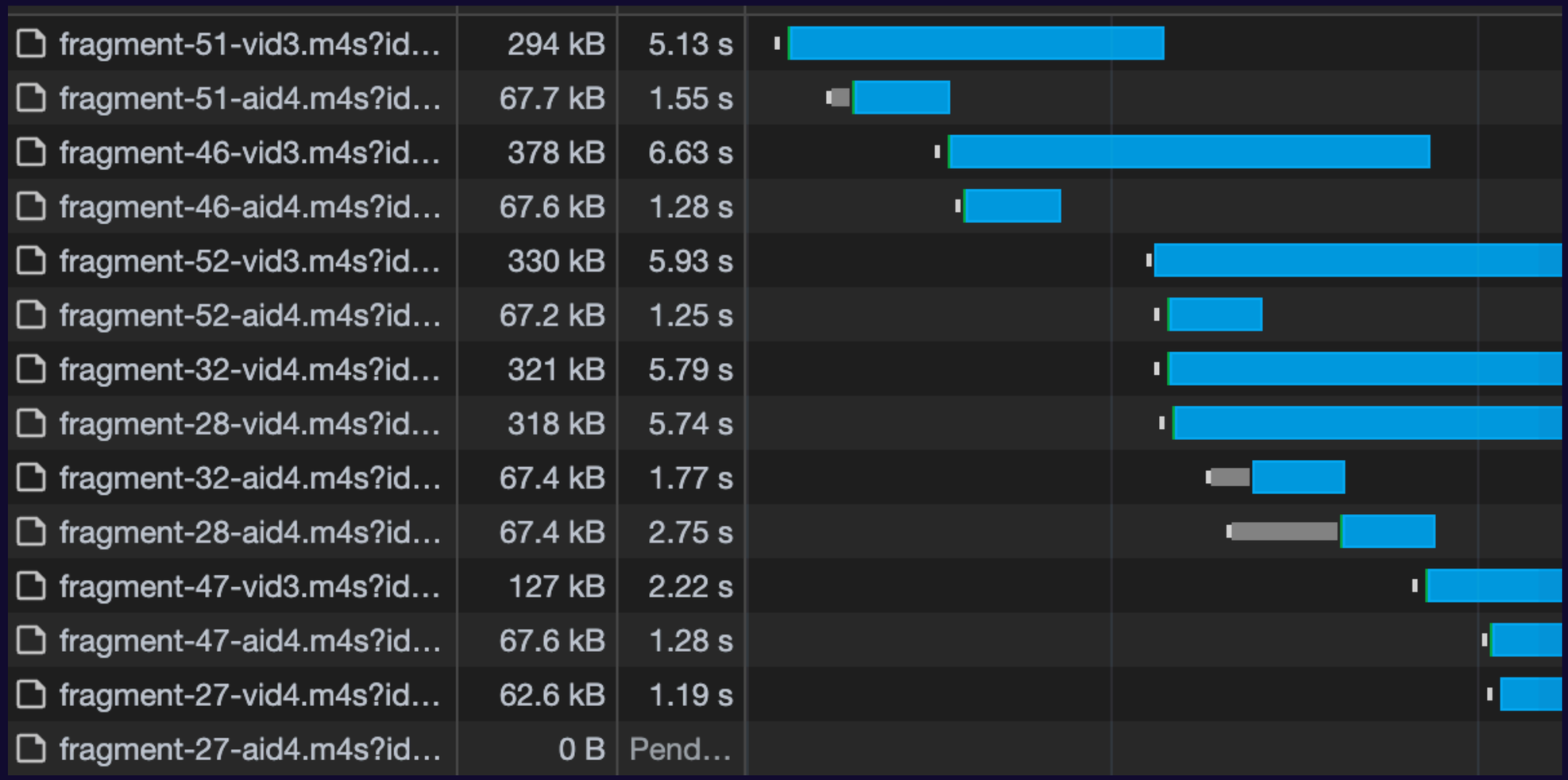
Это плохо

- Медленная сеть
- Ошибаемся в оценке
- Занижаем качество видео
- Ошибаемся еще больше

Делаем еще хуже там, где все и так плохо

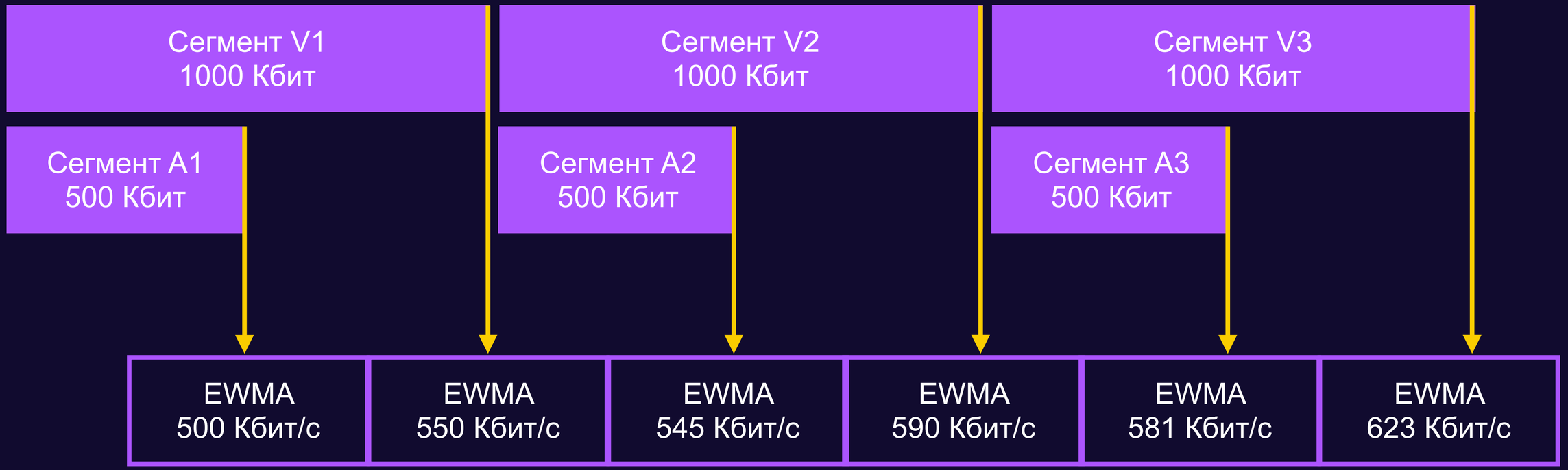
Если потоков больше, все совсем грустно

Как такое считать?

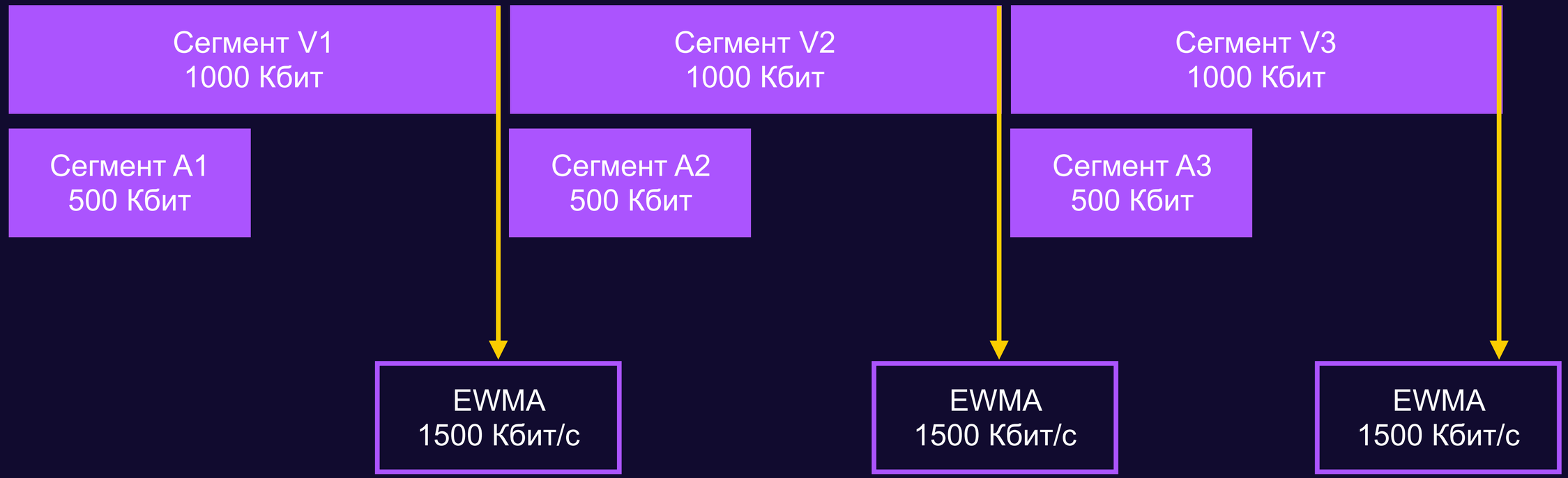


Нужна модель для произвольного N потоков

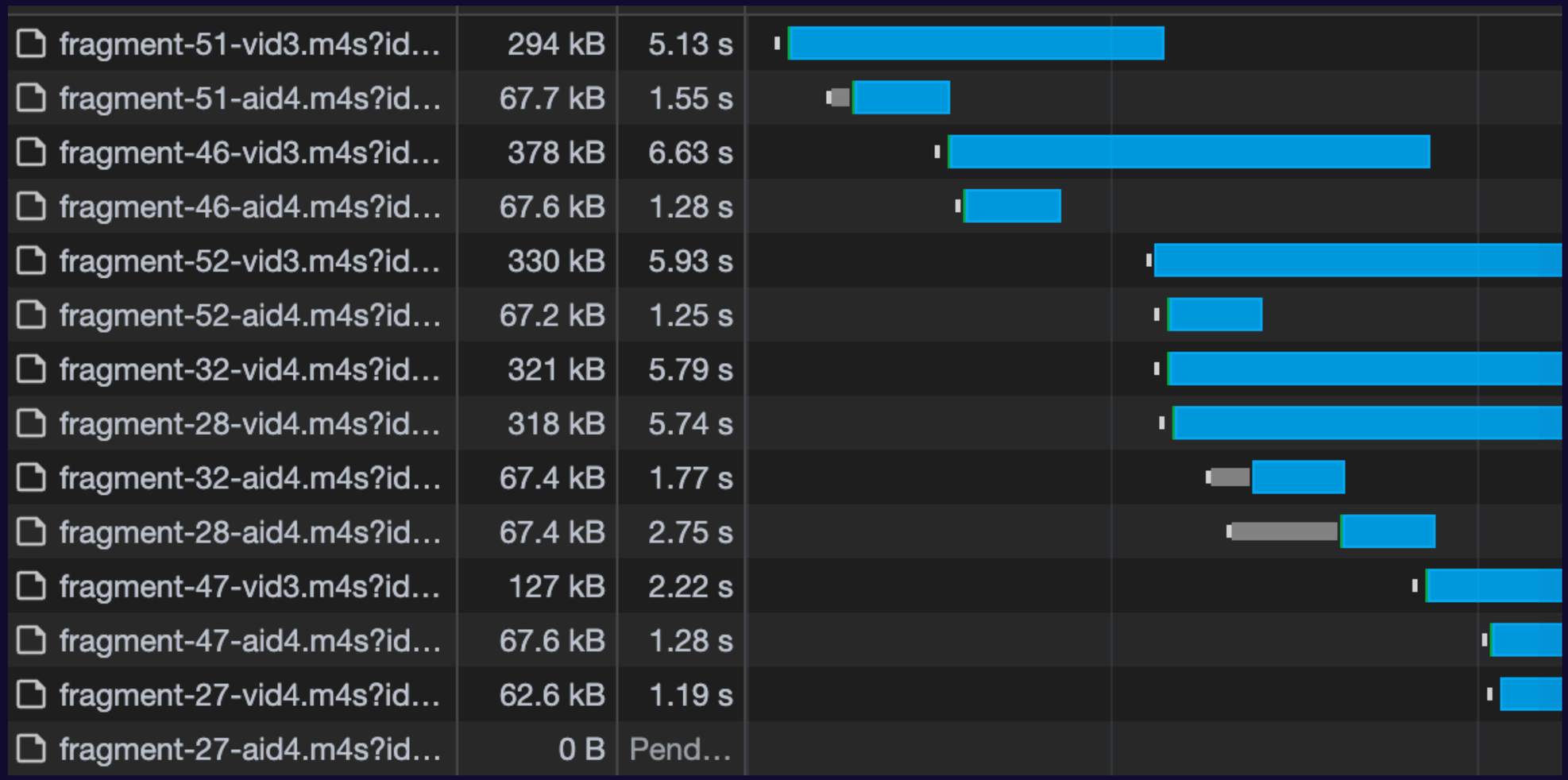
Вернемся к схеме



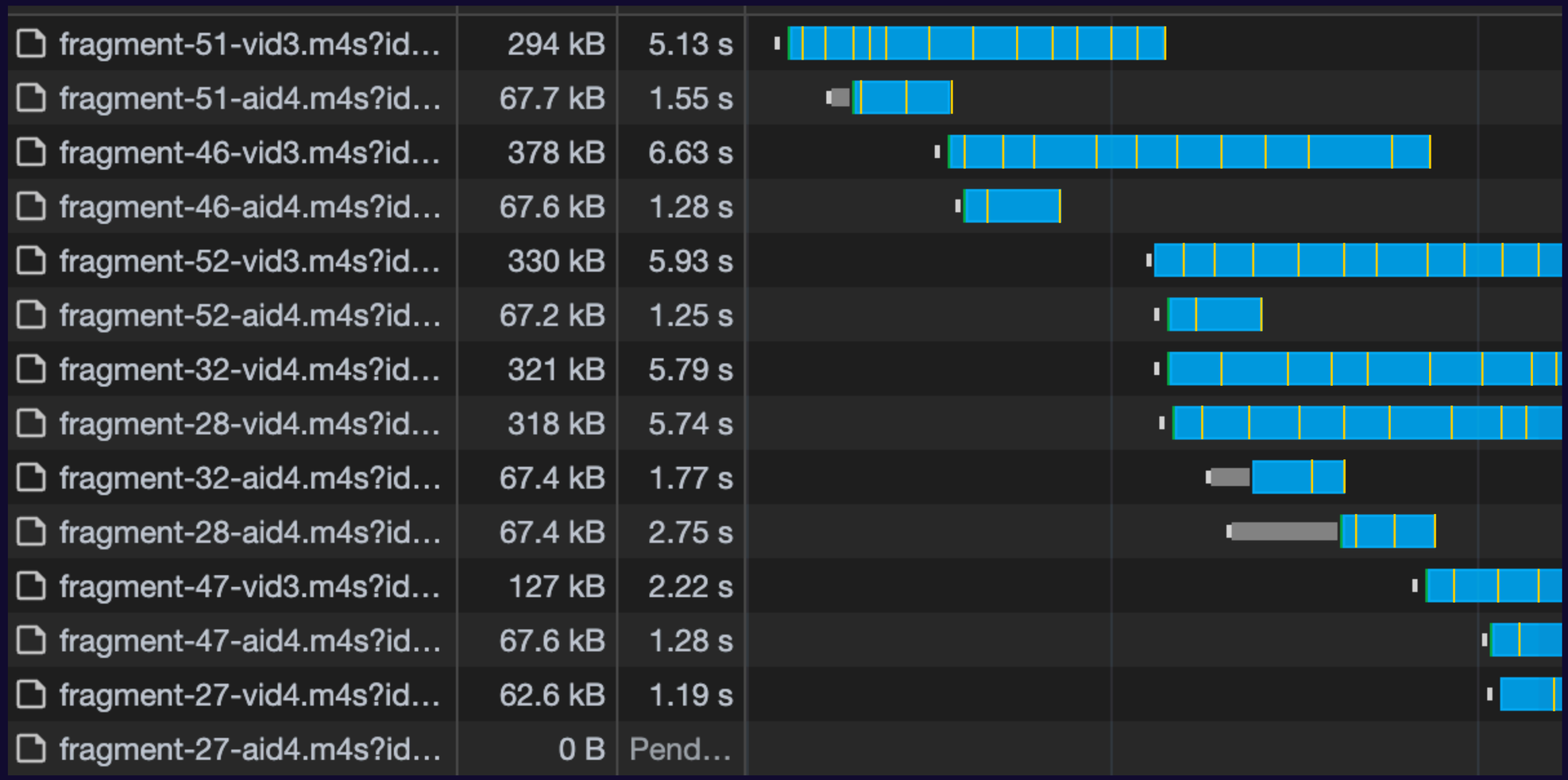
Хотим вот так



Всегда будет живой запрос



Всегда будет живой запрос



Внутри плеера

network_layer

ewma_bandwidth_estimator

Внутри плеера

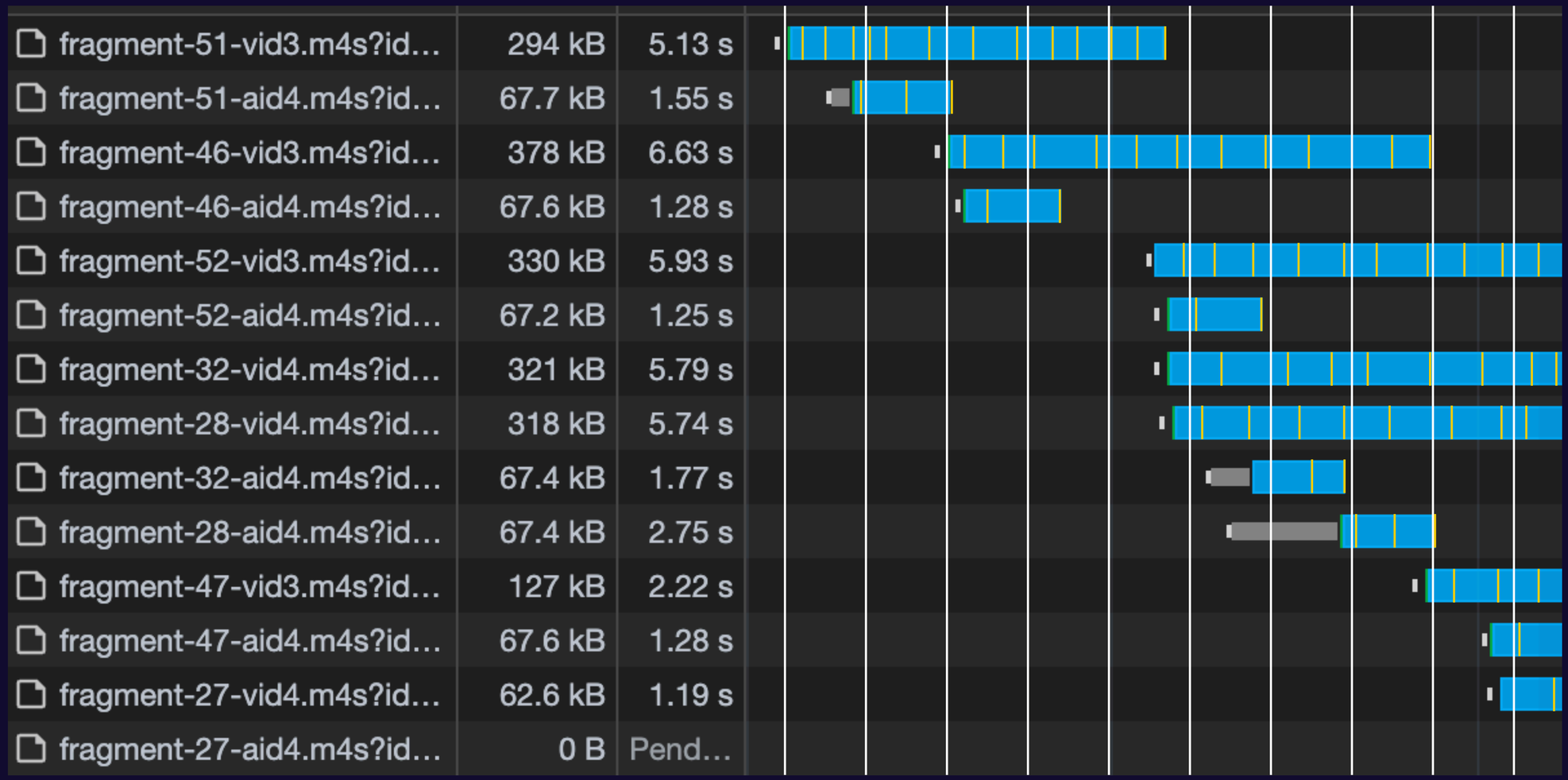
Достаточно научиться ждать

network_layer

ewma_bandwidth_estimator

Достаточно научиться ждать

Примерно так



Дискретизация

Процесс измерения значения сигнала через определенные промежутки времени

Сигнал — объём загруженных данных

Как закодить?

Вот такой плеер

```
class Player {
  constructor() {
    this.networkLayer = new NetworkLayer();
    this.bwEstimator = new BWEstimator();
    this.abr = new ABR(this.bwEstimator);

    this.networkLayer.on('Progress',
      (size /* Bits */, time /* Seconds */) =>
        this.bwEstimator.sample(size, time));
  }
}
```

Сетевой слой

```
class NetworkLayer extends EventEmitter {
  download(url) {
    const request = new Request(url);
    request.on(‘Progress’, (size /* Bits */) => {
      this.emit(‘Progress’, size, getTimeFromPreviousProgress());
    });
  }
}
```

Вводим новую сущность

```
class ProgressAccumulator extends EventEmitter {
  requestProgress(size) {
    this.accumulatedSize += size;
  }
  requestStarted() {
    setIntervalIfNotExist(this.handleInterval);
  }
  requestFinished() {
    clearIntervalIfAllRequestsFinished();
  }
}
```

handleInterval

```
handleInterval() {
  const time = getTimeFromPreviousCall();
  const size = this.accumulatedSize;
  this.accumulatedSize = 0;
  this.emit('Progress', size, time);
}
```

Немного пропатчим плеер

```
class Player {
  constructor() {
    // . . .
    this.pa = new ProgressAccumulator();
    this.networkLayer.on('Progress',
      (size, time) => this.pa.sample(size, time));

    this.pa.on('Progress',
      (size, time) => this.bwEstimator.sample(size, time));
  }
}
```

Fork с правками в shaka-player

https://clck.ru/36eL24

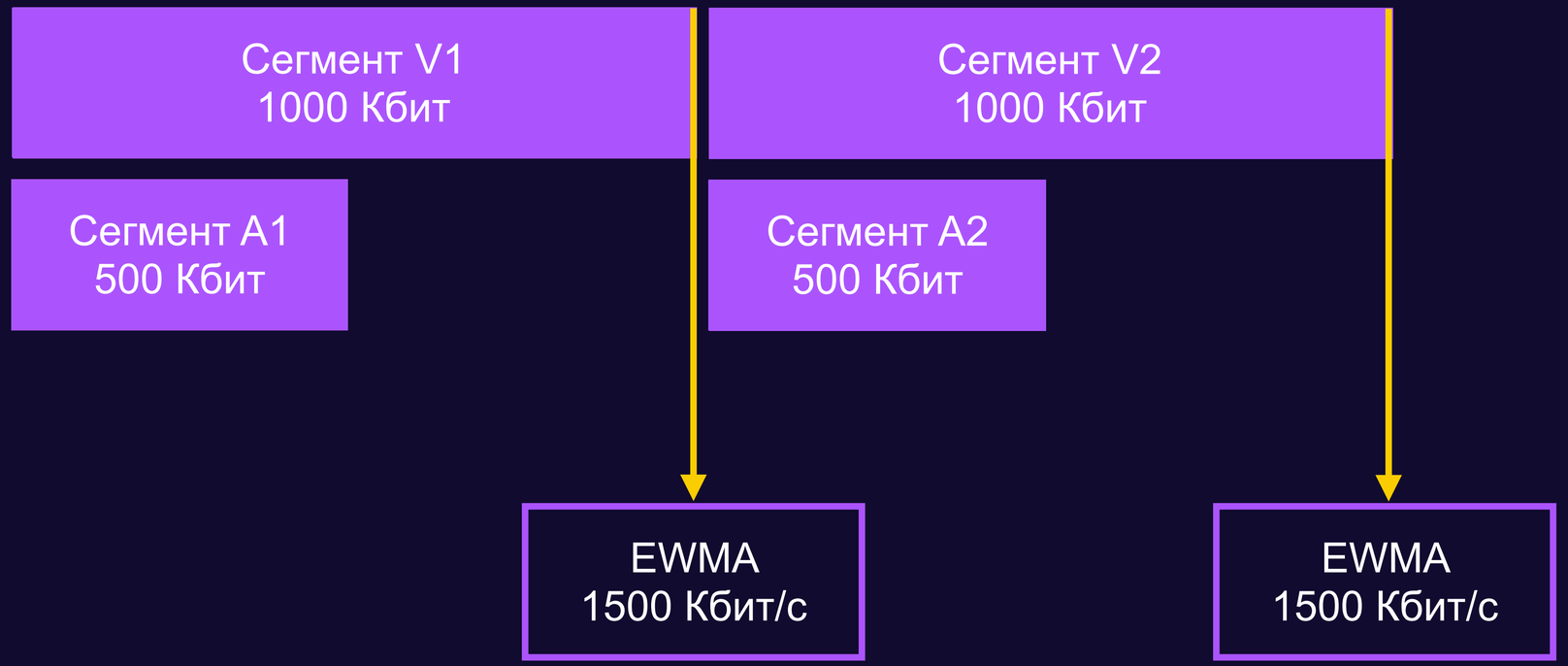
На бумаге хорошо,
а работает ли?

Не у каждого пользователя локальный видео-сервер

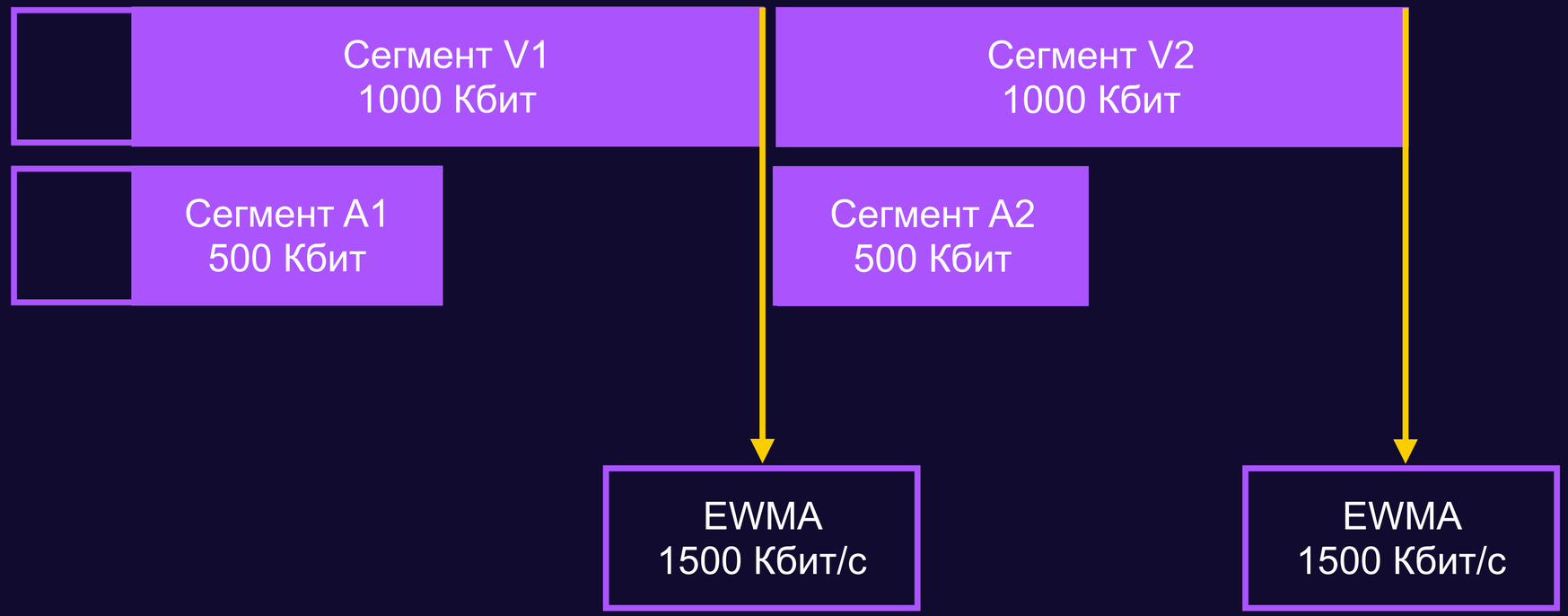
Добавим реализма

Ничего не работает

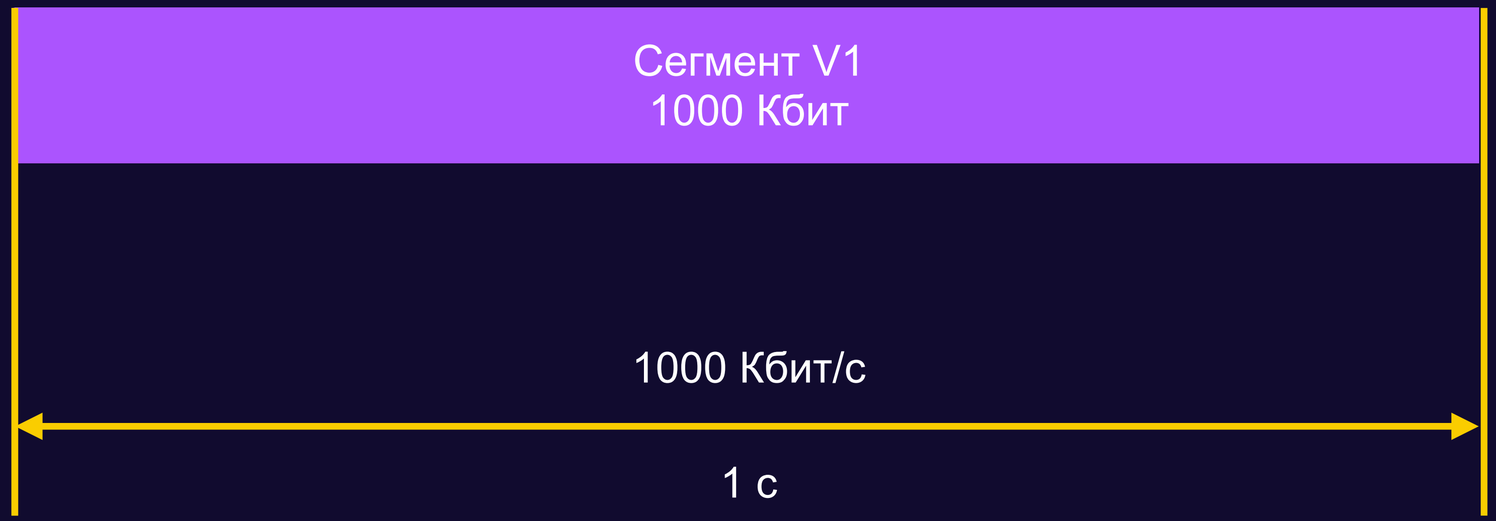
Дело в задержках (или TTFB)



Дело в задержках (или TTFB)



Как задержки влияют



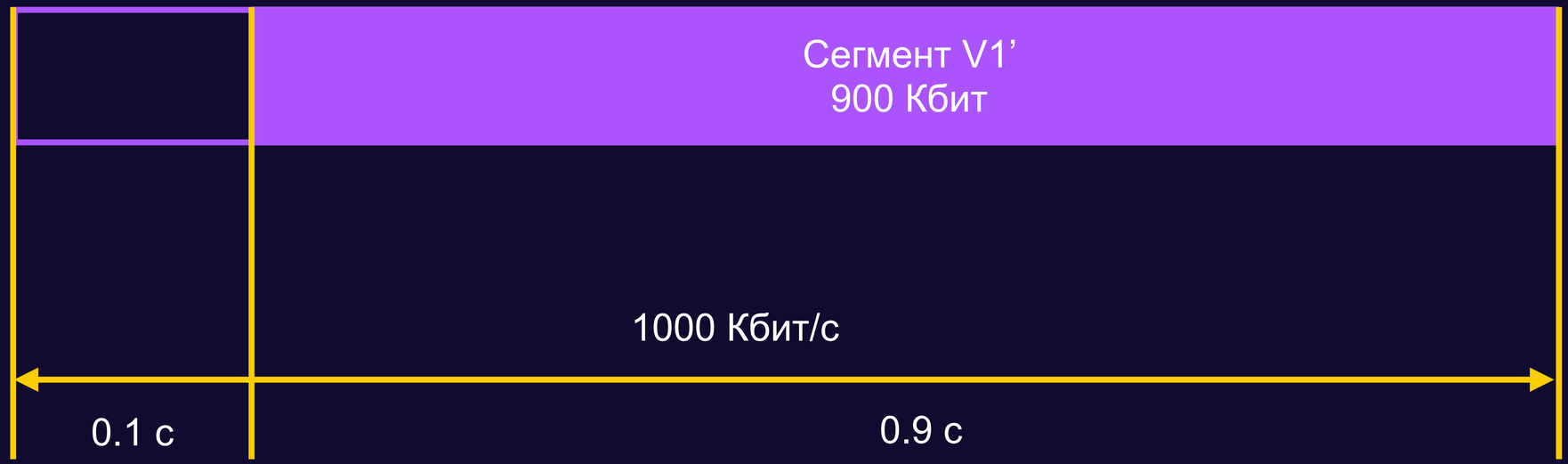
Как задержки влияют

Как задержки влияют

Нужно учитывать TTFB

Качать сегменты за время
SegmentDuration – TTFB

Качать сегменты за время
SegmentDuration – TTFB



$$EffectiveBW = RealBW/SD * (SD - TTFB)$$

Очень упрощенная версия кода shaka-player

```
class AbrManager {
  chooseVariant() {
    const bw = this.getBandwidthEstimate();
    let chosen = variants[0]; // Самое плохое качество
    for (const variant of variants) {
      if ((variant.bandwidth / 0.95) < bw) {
        chosen = variant;
      }
    }
  }
}
```

Добавим логики для TTFB

```
class AbrManager {
  chooseVariant() {
    const bw = this.getBandwidthEstimate();
    const ttfb = this.getTtfbEstimate();
    let chosen = variants[0];
    for (const variant of variants) {
      const effBw = bw / variant.duration * (variant.duration - ttfb);
      if ((variant.bandwidth / 0.95) < effBw) {
        chosen = variant;
      }
    }
```

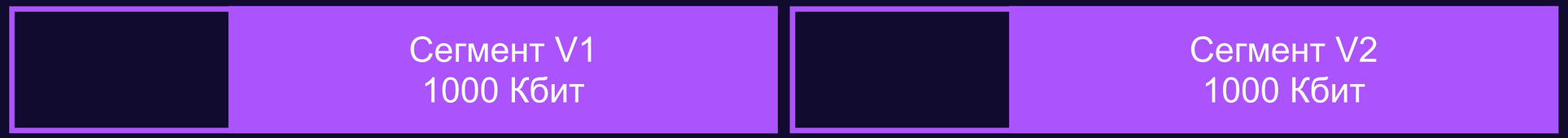
Пришли к тому,
с чего начали

Что по качеству?

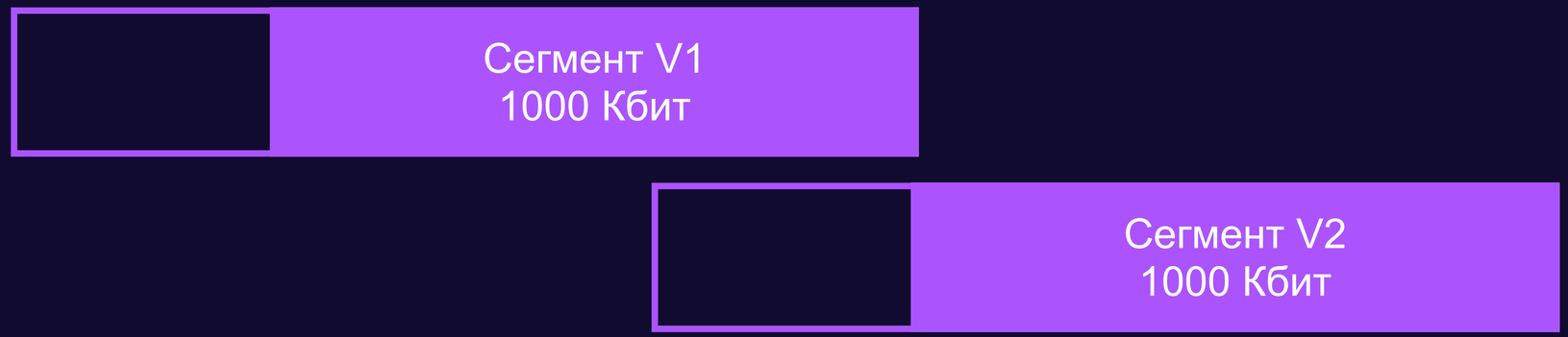
Высота видео	Время смотрения до	Время смотрения после	Изменение в ПП
144p	0.4%	0.2%	- 0.2%
240p	1.3%	1.2%	- 0.1%
360p	11.7%	9.8%	- 1.9%
480p	11.1%	10.3%	- 0.8%
720p	47.4%	49.9%	+ 2.5%
1080p	28.1%	28.6%	+ 0.5%

$$EffectiveBW = RealBW/SD * (SD - TTFB)$$

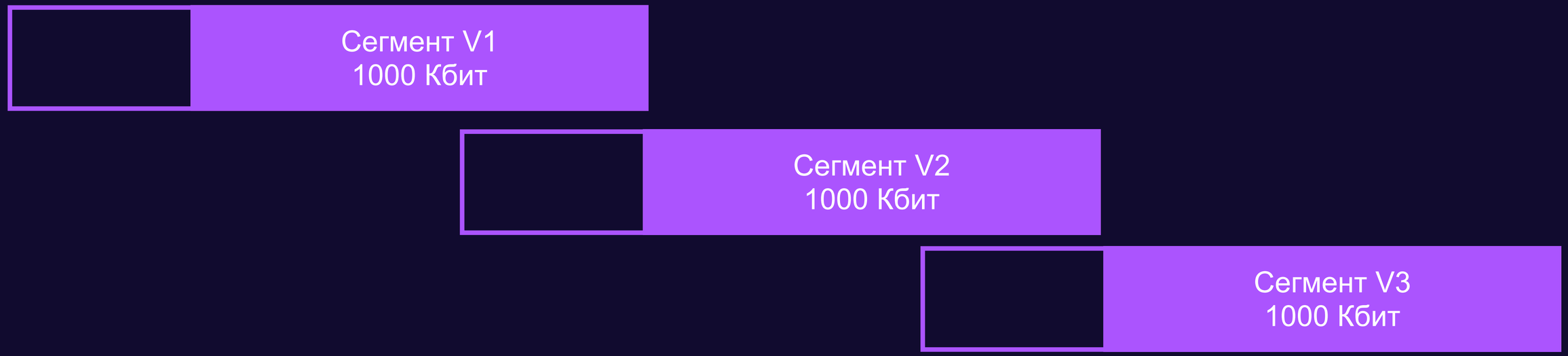
Можно лучше



Можно лучше



Можно лучше


Сегмент V1
1000 Кбит
Сегмент V2
1000 Кбит
Сегмент V3
1000 Кбит

Не везде можно сделать

Одна из проблем

И снова качество

Высота видео	Время смотрения до	Время смотрения после	Изменение в ПП
144p	-	-	-
240p	0.03%	0.03%	-
360p	4.2%	4.1%	- 0.1%
480p	6.3%	6.2%	- 0.1%
720p	50.9%	50.9%	-
1080p	38.6%	38.8%	+ 0.2%

-2.5%

Количества буферизаций

-0.3%

Средней длительности буферизации

Вместо послесловия

Знайте как работают базовые слои

Если что-то ведет себя не так,
как ожидаете — копайте

Если делаете плеер — настоятельно
рекомендую ознакомиться с форком

Не забудьте обновиться

TG: Страдания юного видеоинженера

https://clck.ru/36bqF8

Infrastructure

Буду рад ответить на ваши вопросы

Константин Петряев
Yandex Infrastructure