Перформанс метрики

Что сегодня будет

Расскажу про

- Клиент-серверная система
- latency/throughput/bandwidth
- median, p75, p90
- Браузерный перформанс
- Как на все это смотрит пользователь

Клиент-Серверная инфраструктура

Клиентская машина

Наша великая инфраструктура

Наш мощный сервер

Наши приложения - “Data Intensive”

Клиентская машина 1

Клиентская машина 2

Наши провайдеры

Наш мощный сервер

Наша великая сеть датацентра

Мощный сервер БД

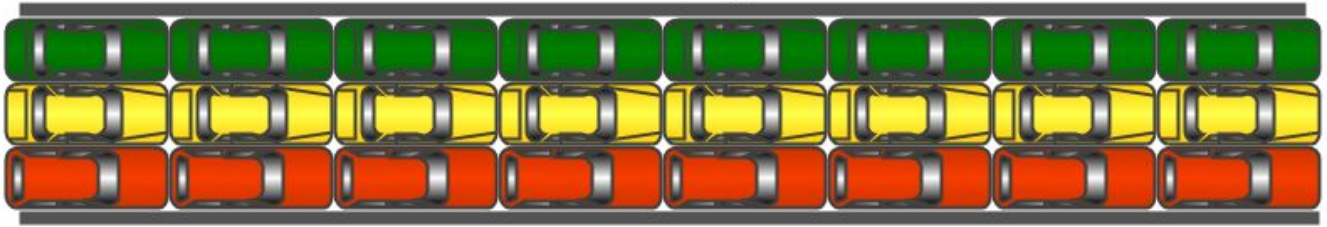
Мощный сервер Чего то еще

Наши сервисы большую часть времени проводят в ожидании ответа от других сервисов

Великолепная тройка

Ширина канала

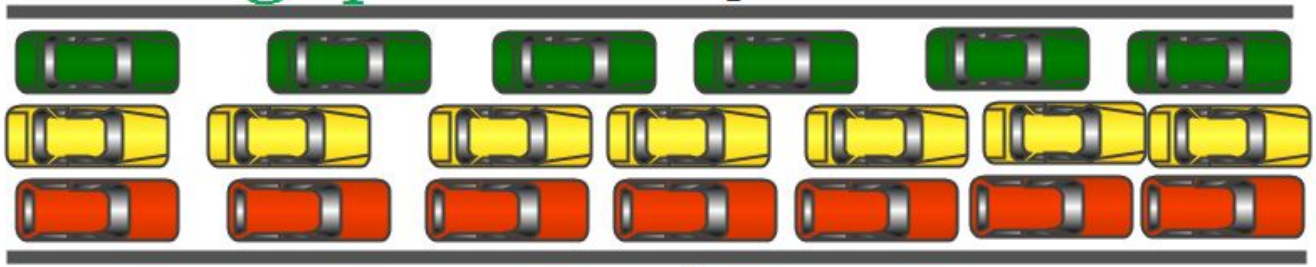
Bandwidth 24 Cars per second



= RPS

Пропускная способность

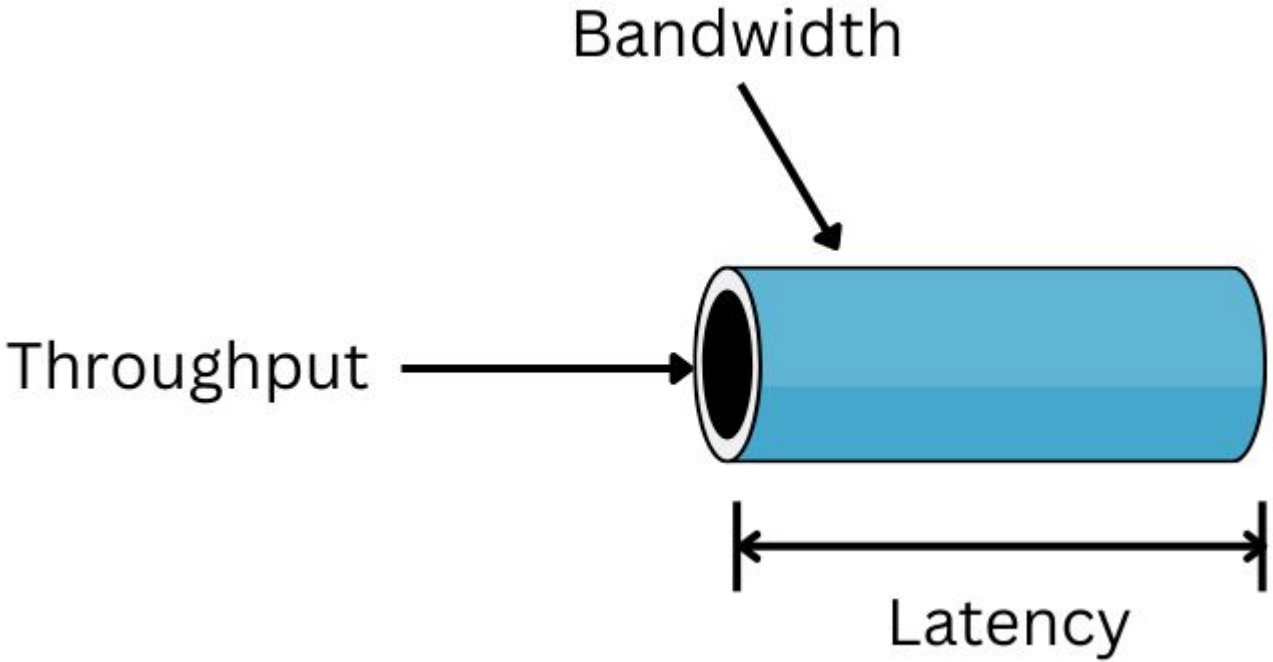
Throughput 20 Cars per second

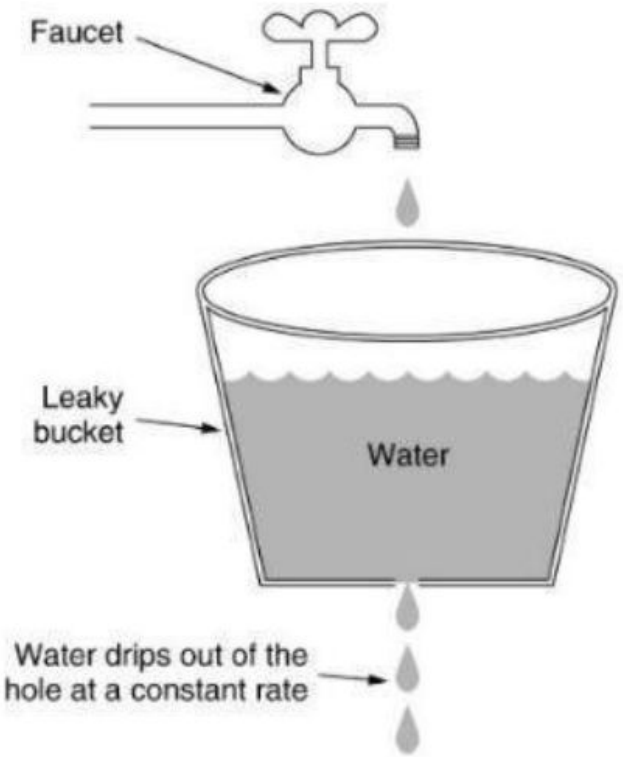


+

Latency - Длина “пути”

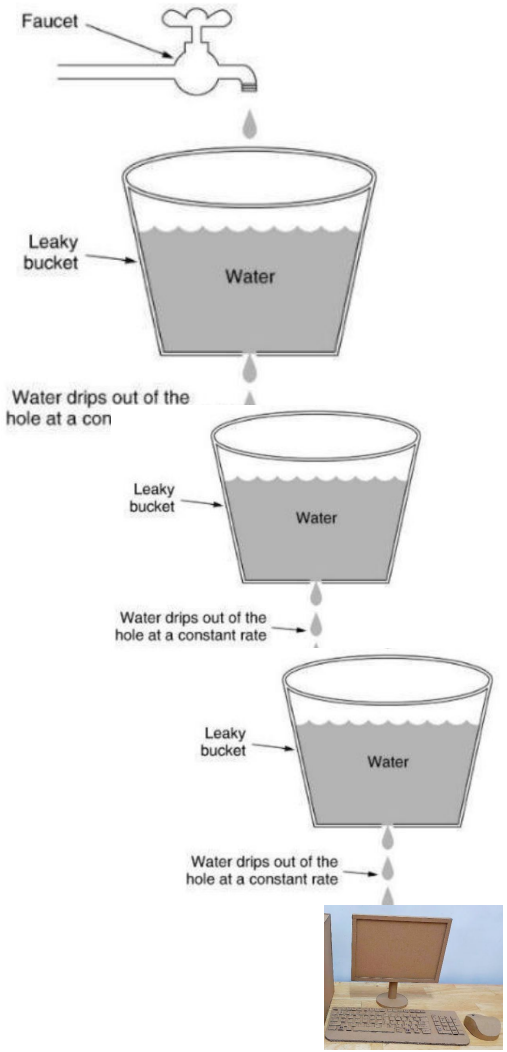
Аналогия с шлангом





Входной поток != Скорость выполнения запроса

Вывод – Где то есть очередь обработки



На практике

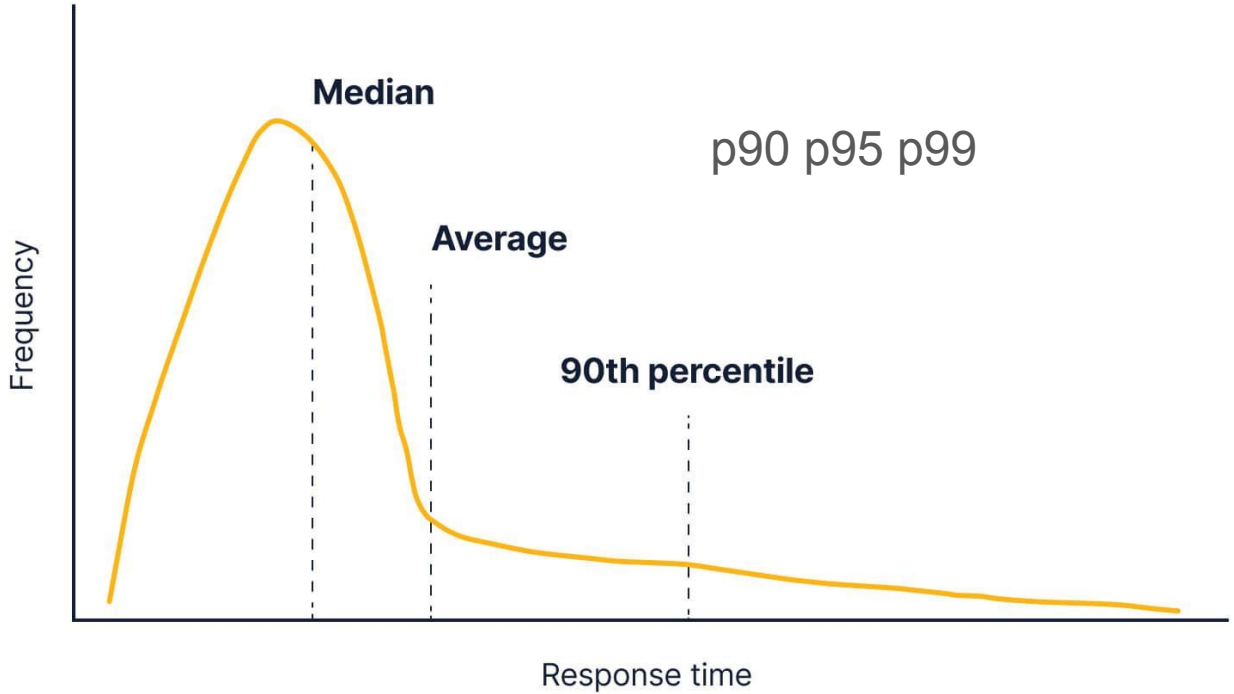
Очередей целая цепочка

Некоторые “бочки” - самые **медленные**

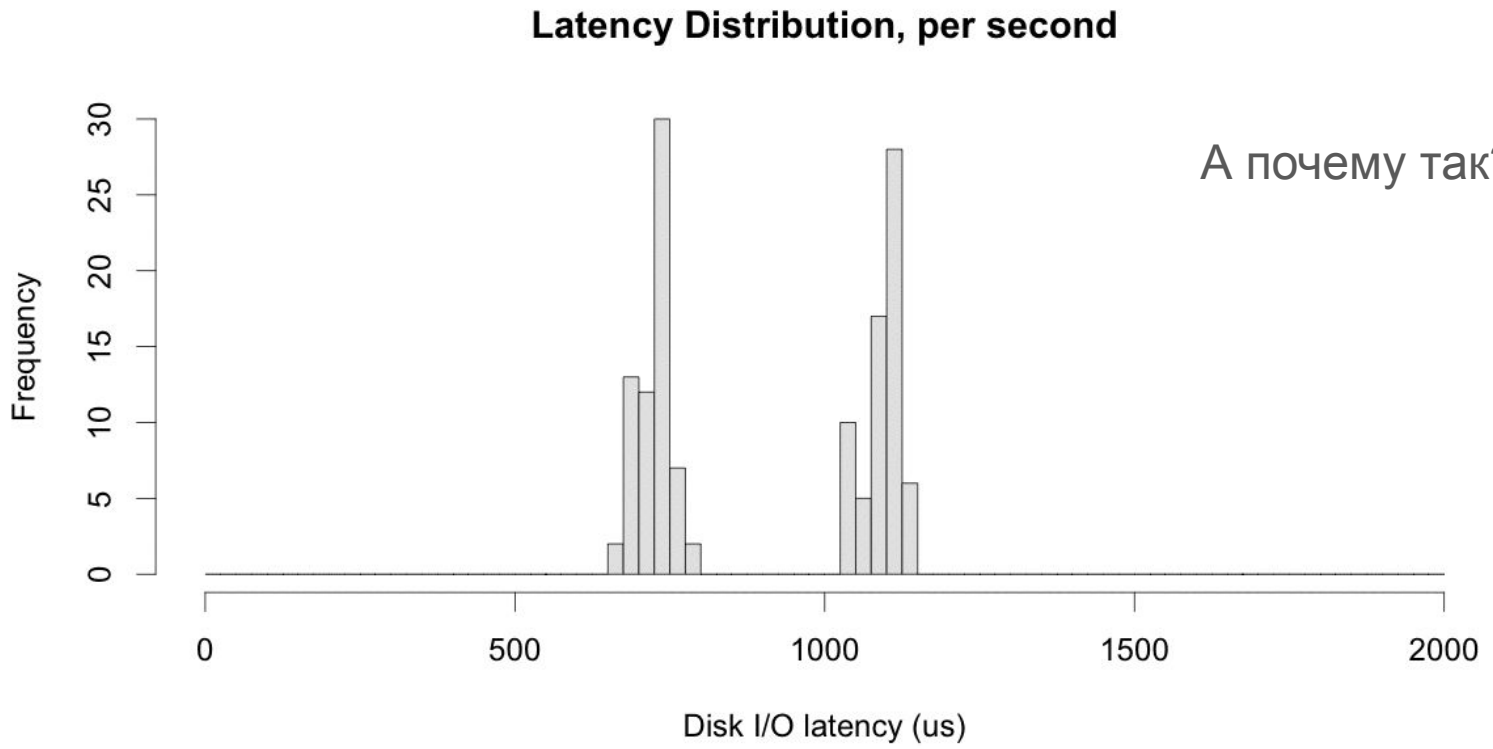
(бутылочные горлышки)

Обработка запросов моделируется хорошо системой очередей

Среднее никому не интересно, всем интересен хвост

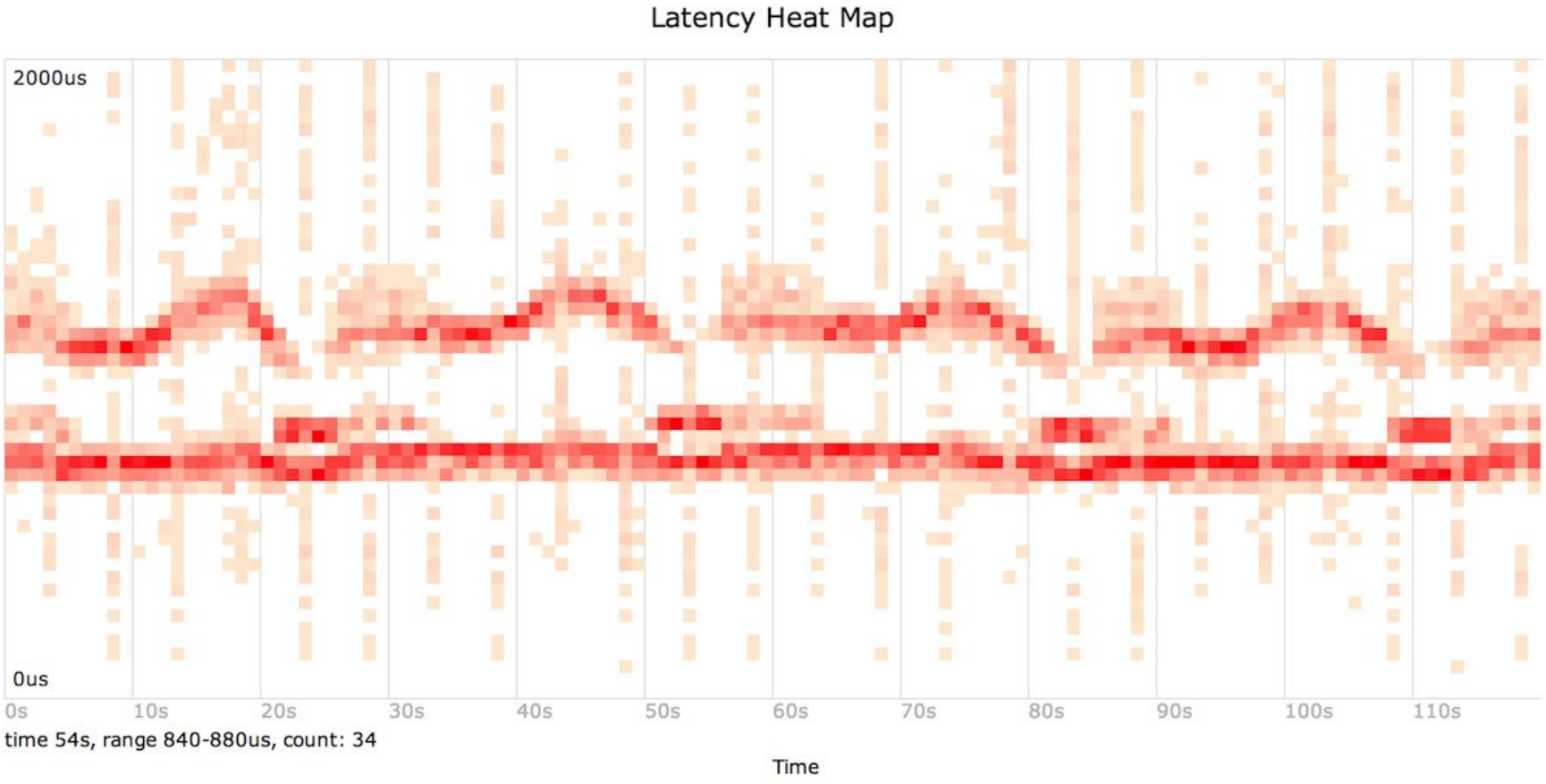


Latency на практике

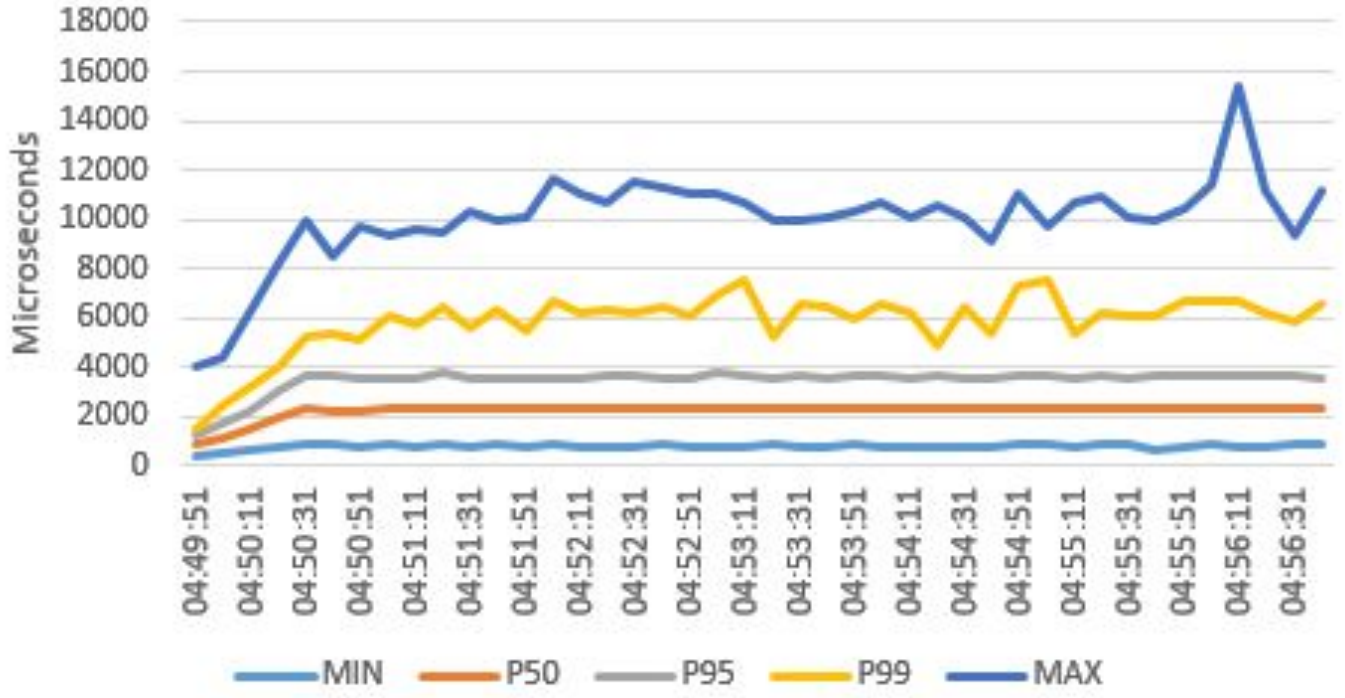


А почему так?

Тепловая карта



Latency на графиках



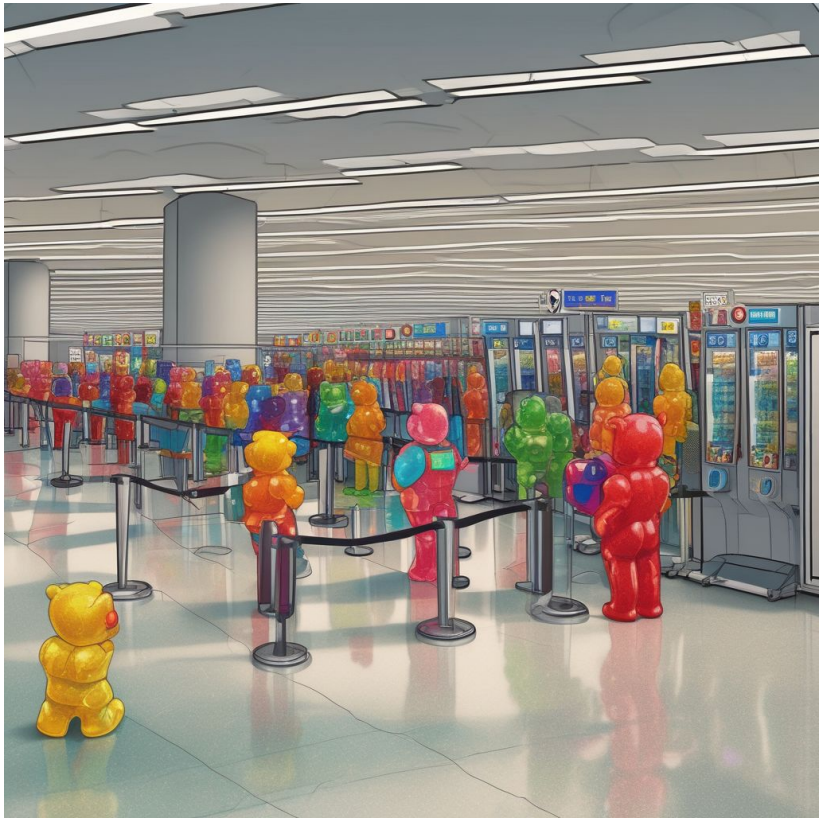
Конфликт требований к оптимизации

- Для того, чтобы снизить latency, необходимо уменьшить среднюю длину очереди (очередей) в системе
- Для того, чтобы увеличить RPS (throughput), необходимо увеличить среднюю длину очередей в системе, чтобы избежать простоев на фоне неравномерной нагрузки
- Для равномерного распределения нагрузки в распределённой системе, необходимо, чтобы очереди к разным узлам были примерно одного размера

Под нагрузкой все становится медленным

Как сделать быстрее

- Ищи **узкое** горлышко
- Под ***нагрузкой*** в какой то момент все становится сильно *хуже*, делай ***нагрузочные***
- Не используй ***среднее***
- Смотри на распределение
- *Оптимизировать* под latency и throughput это разные вещи
- ***Умей*** бенчить



Графана, и что в ней встретишь

- метрики **сервисов**,
- метрики **серверов**
- иногда бизнес **метрики**

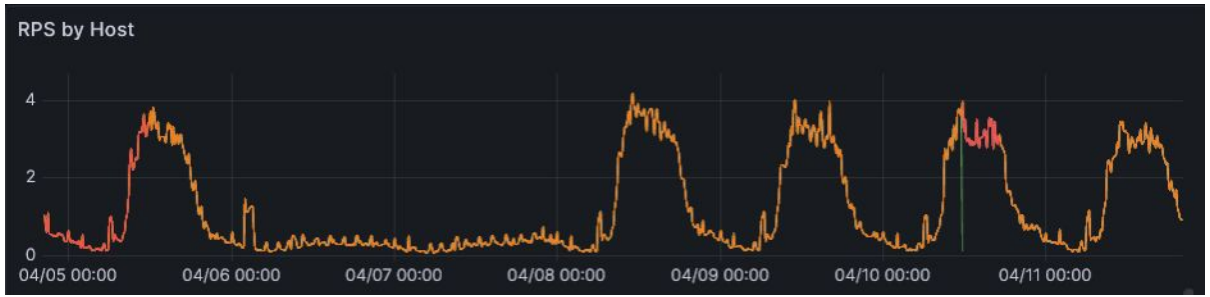
Полезное

- Latency (p95)
- RPS = *Throughput*
- **Thread pool size**
- CPU load = *Bandwidth*
- *Длина/Лаг очереди*

Типичный график дневной нагрузки

- Когда меряешь перформанс нужно учитывать

  что люди не всегда на **работе**

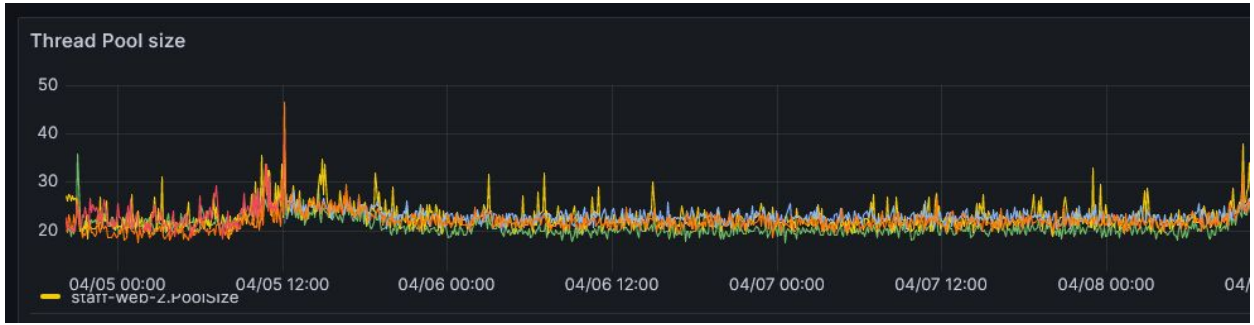


- Когда сервис не нагружен, перформанс оценить ***нереально***
- Лучше всего, сравнивать с **прошлым** аналогичным периодом

Thread Pool size

- Очень часто сервис ждет ответа от кого то еще

  (например от базы данных)



  Если сервис написан криво поток запроса лочится и ждет ответа

  Ради чего придумали (async/await) - dotnet поднимает доп потоки если все чем то заняты (или ждут)

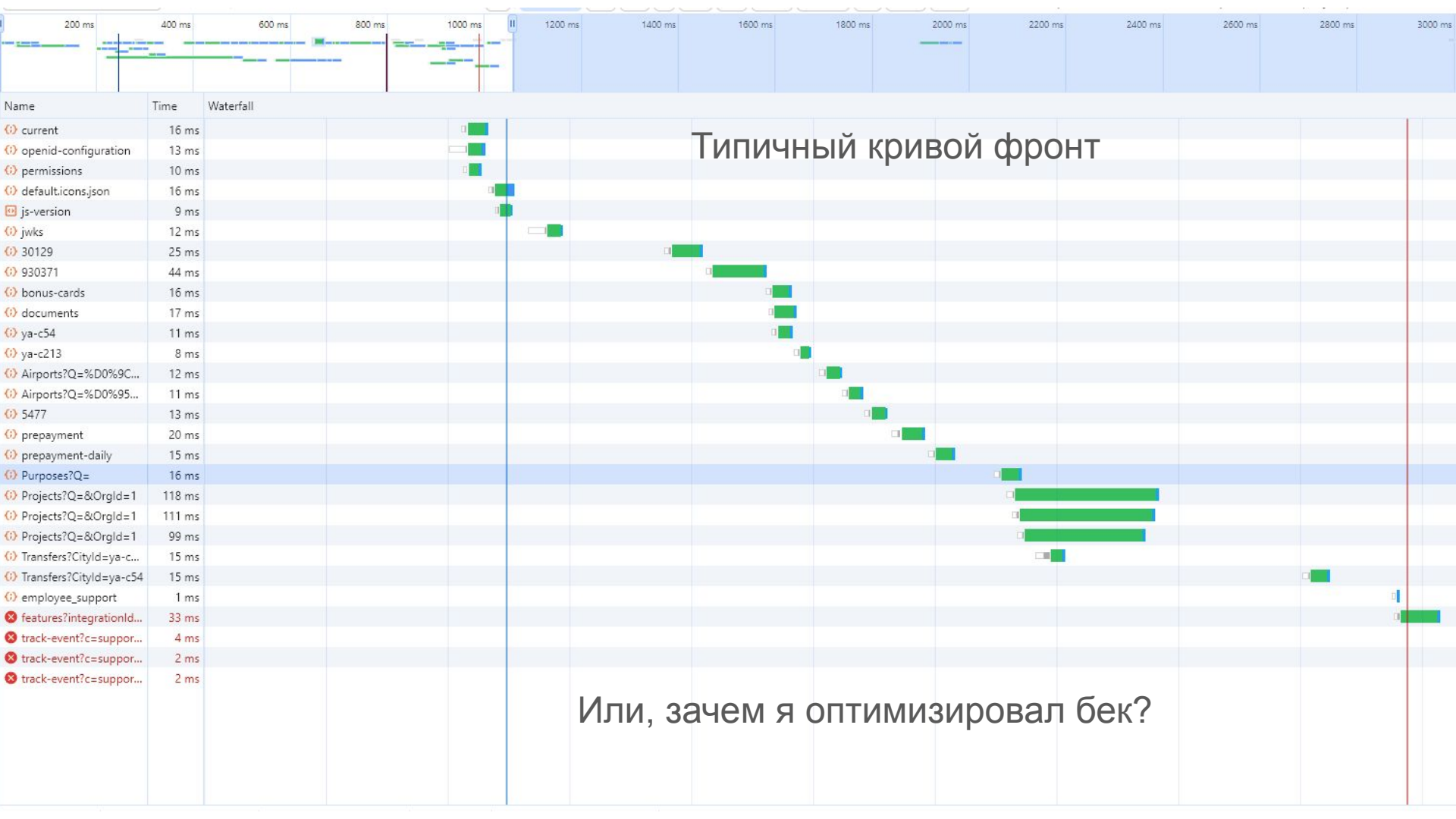
- На графике можно увидеть проблему с блокировками, с кривым async-await

GC Time

- Важно для сервисов где есть уборка мусора
- Требует отдельной заботы

Браузерный перформанс

Кривой фронт может похоронить все оптимизации на сервере и в инфраструктуре


200 ms
400 ms
600 ms
800 ms
1000 ms
1200 ms
1400 ms
1600 ms
1800 ms
2000 ms
2200 ms
2400 ms
2600 ms
2800 ms
3000 ms
Name
Time
Waterfall
current 16 ms
openid-configuration 13 ms
permissions 10 ms
default.icons.json 16 ms
js-version 9 ms
jwks 12 ms
30129 25 ms
930371 44 ms
bonus-cards 16 ms
documents 17 ms
ya-c54 11 ms
ya-c213 8 ms
Airports?Q=%D0%9C... 12 ms
Airports?Q=%D0%95... 11 ms
5477 13 ms
prepayment 20 ms
prepayment-daily 15 ms
Purposes?Q= 16 ms
Projects?Q=&OrgId=1 118 ms
Projects?Q=&OrgId=1 111 ms
Projects?Q=&OrgId=1 99 ms
Transfers?CityId=ya-c... 15 ms
Transfers?CityId=ya-c54 15 ms
employee_support 1 ms
features?integrationId... 33 ms
track-event?c=suppor... 4 ms
track-event?c=suppor... 2 ms
track-event?c=suppor... 2 ms
Типичный кривой фронт
Или, зачем я оптимизировал бек?

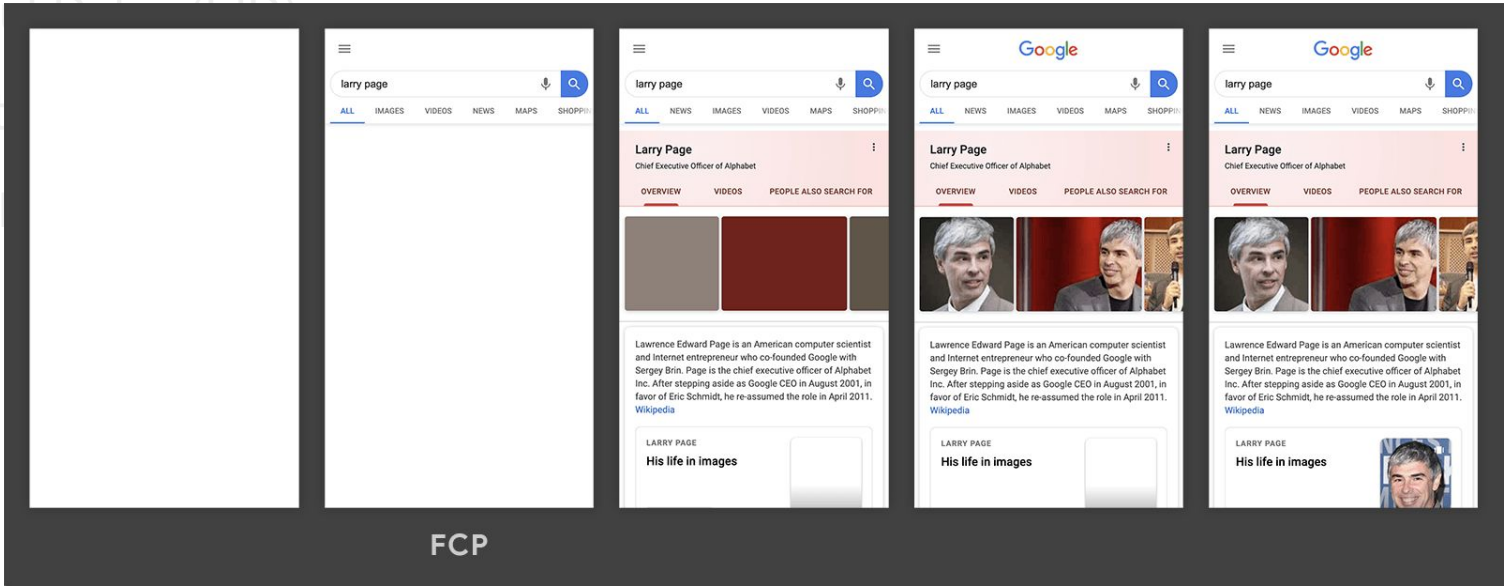
Как измеряют перформанс в браузерах

First Contentful Pain (FCP)

First Inp

Cumulat

Time to



Как измеряют перформанс в браузерах

First Contentful Pain (LCP)

First Input Delay (FID)

Время реакций на ввод, инпут лаг

Cumulative Layout Shift (CLS)

Time to First Byte (TTFB)

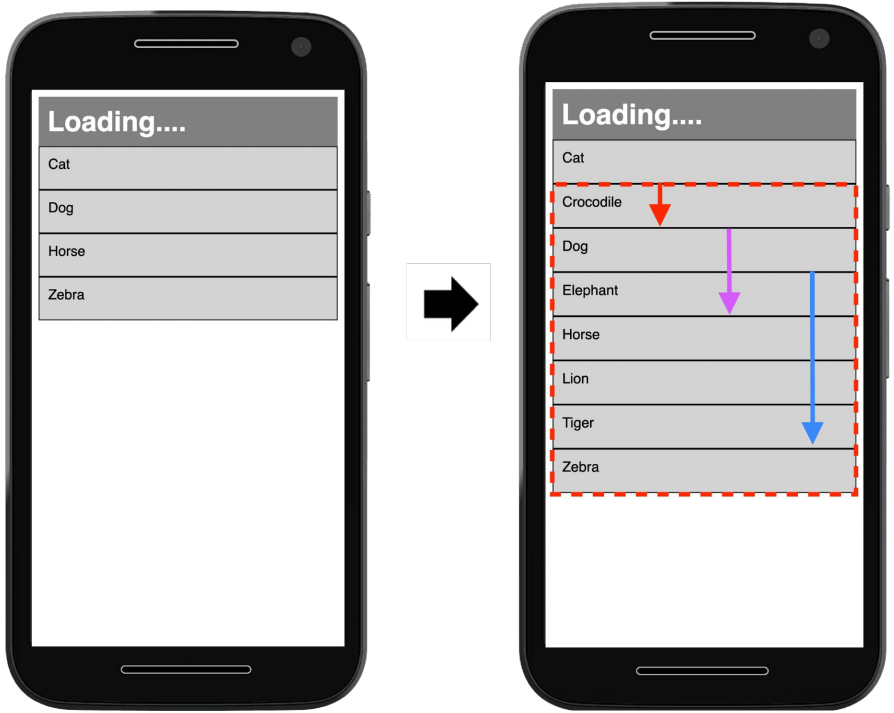
Как измеряют перформанс в браузерах

First Contentful Pain (LCP)

First Input Delay (FID)

Cumulative Layout Shift (CLS)

Time to First Byte (TTFB)



Когда сайт грузится, интерфейс скачет

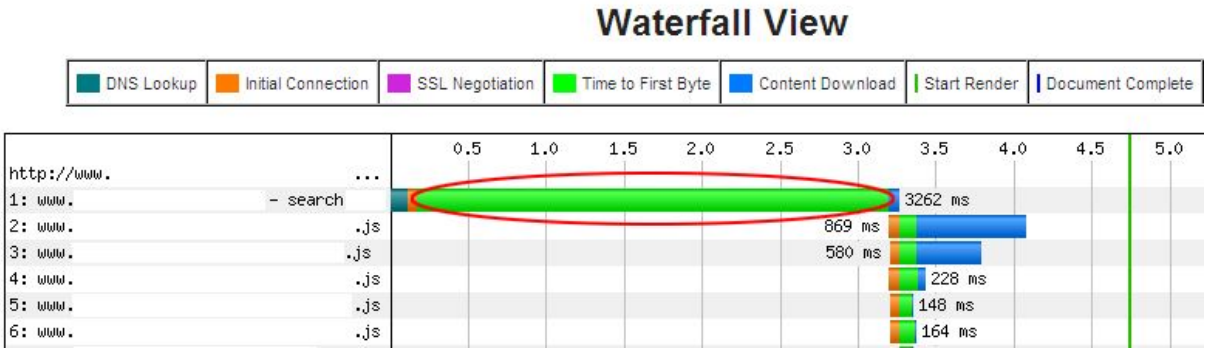
Как измеряют перформанс в браузерах

First Contentful Pain (LCP)

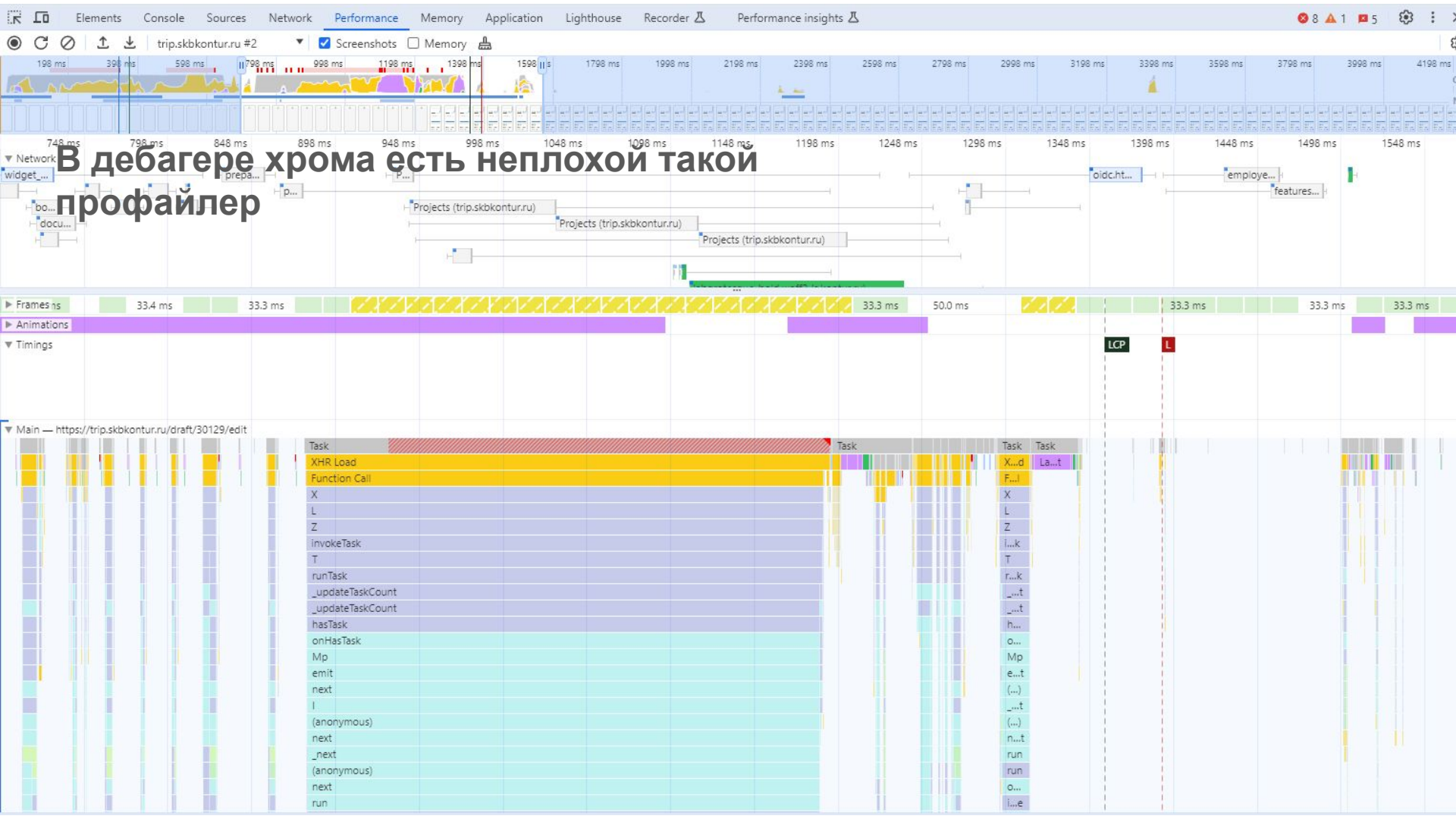
First Input Delay (FID)

Cumulative Layout Shift (CL

Time to First Byte (TTFB)



Latency со стороны клиента
время до первого байта от сервера



В дебагере хрома есть неплохой такой профайлер

А Пользователь?

- Больше 300мс заметно

- Делай **лоадер** если дольше, но лучше чини

- Не у всех хороший инет, **лоадер** нужен даже если сервер сегодня быстрый

Конец

Вопросы?