

# LLM под нагрузкой

Как измерять производительность  
self-hosted моделей



**Роман Песков,**  
Старший инженер-разработчик,  
AI-платформа



# Кому полезен доклад



## Работаете с LLM как с сервисом

Поймёте, что стоит за словами «модель тормозит», и какие числа спрашивать у инфраструктуры



## MLOps и ML-инженеры

Получите порядок измерений: метрики времени ответа, нагрузочный тест, сколько карт брать

Дослушайте до конца и заберите с собой **чеклист**

# Циан – ведущая цифровая платформа операций с недвижимостью



**20,2** млн

уникальных  
пользователей в месяц

**2,1** млн

актуальных  
объявлений

**99%**

чистота базы

**25** лет

экспертизы

**15,2** млрд руб

выручка за 12  
месяцев 2025 года  
выросла  
на 16,7%

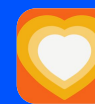
**Топ-15**

крупнейших сайтов  
недвижимости в  
мире  
по версии Similar  
Web

**1 000+**

сотрудников

**№1**



proptech в инклюзии:  
~6 тыс. новых объявлений  
в день содержат  
параметры, важные для  
людей на колясках

# Зачем нам self-hosted



## Данные

Персональные данные и внутренние тексты не уходят к внешним провайдерам.



## Стоимость

Для задач с высокой загрузкой свой инференс может быть дешевле API.



## Контроль

Свои модели, версии и процесс обновлений.

Цена self-hosted: все вопросы производительности теперь **наши**

# Команда LLM-инфраструктуры



## О команде

- Тимлид и два MLOps-инженера, включая меня
- Запускаем модели и встраиваем их в сервисы
- Отвечаем за производительность



## Моя роль в команде

- MLOps + Python Dev
- Разрабатываю инструменты и сервисы ML/AI-платформы

Отдельного инженера по нагрузке нет: измерение должно быть **дешёвым и повторяемым**

# Инфраструктура: бот-ассистент



## Ожидаемая нагрузка

60 пользователей

Один вопрос раз в 5 минут

Около 8000 токенов входа

До 400 токенов ответа

**В среднем 12 запросов/мин**



## SLO

Первый токен  $\leq 5$  с

Средняя генерация  $\geq 20$  токенов/с

**Оба условия у  $\geq 95\%$  запросов**

# Сколько нужно GPU

1 GPU · Qwen3.8-27B FP8 · vLLM

Измерили: 80 tok/s

одиночные запросы

1

Сколько одновременно

$$80 \div 20 = 4 \text{ req}$$

2

Сколько времени на  
генерацию

$$400 \div 20 = 20 \text{ s}$$

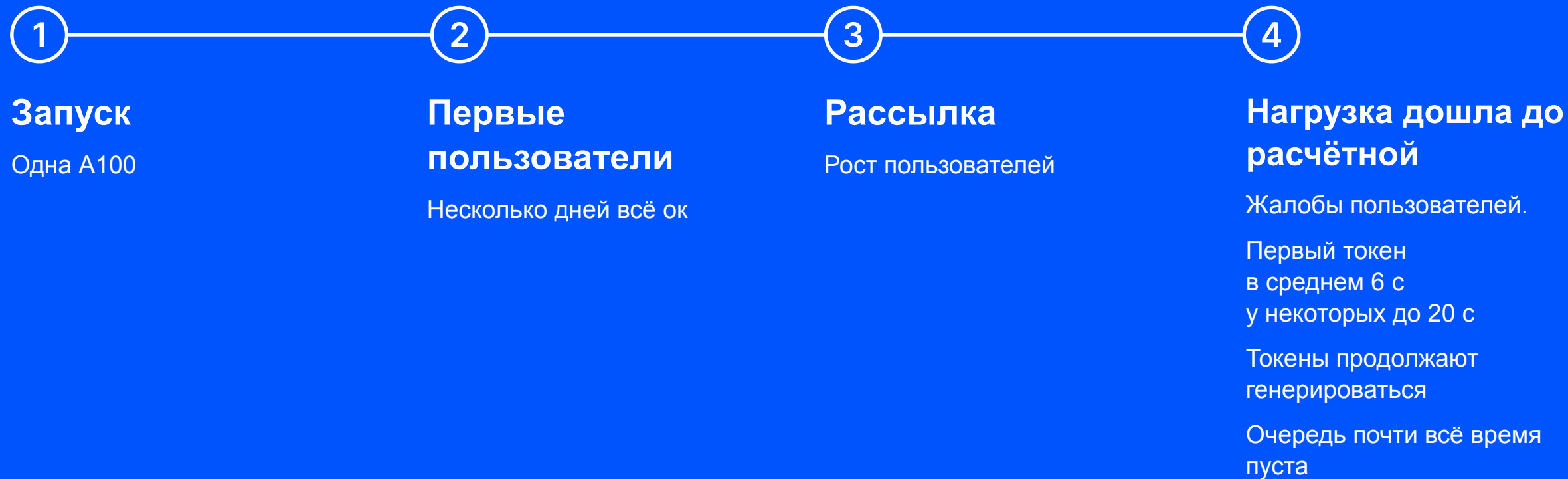
3

Сколько ГОТОВЫХ ОТВЕТОВ В  
минуту

$$4 \times 60 \div 20 = 12 \text{ req/min}$$

Предварительный расчёт обещает необходимые 12 запросов/мин с одной карты

# Запуск



 Что не учёл наш расчёт?

# План доклада

80 tok/s

Какую часть ожидания мы измерили?  
Сценарий 1: время ответа

4 одновременно

Что ограничивает параллельность?  
Сценарий 2: память

12 req/min

Выполняются ли требования при  
таком потоке?  
Сценарий 3: нагрузочный тест

Как повторять измерения  
Паспорт, инструмент, данные, профили



# Сценарий №1



Время ответа



# Внутренний чат-бот

Чат-бот для сотрудников: вопросы, черновики писем, помощь с небольшими текстами.

Тестировали на коротких вопросах-ответах.

1

**T4**  
16 GB

2

**Qwen3 8B · AWQ**  
~6 GB весов

3

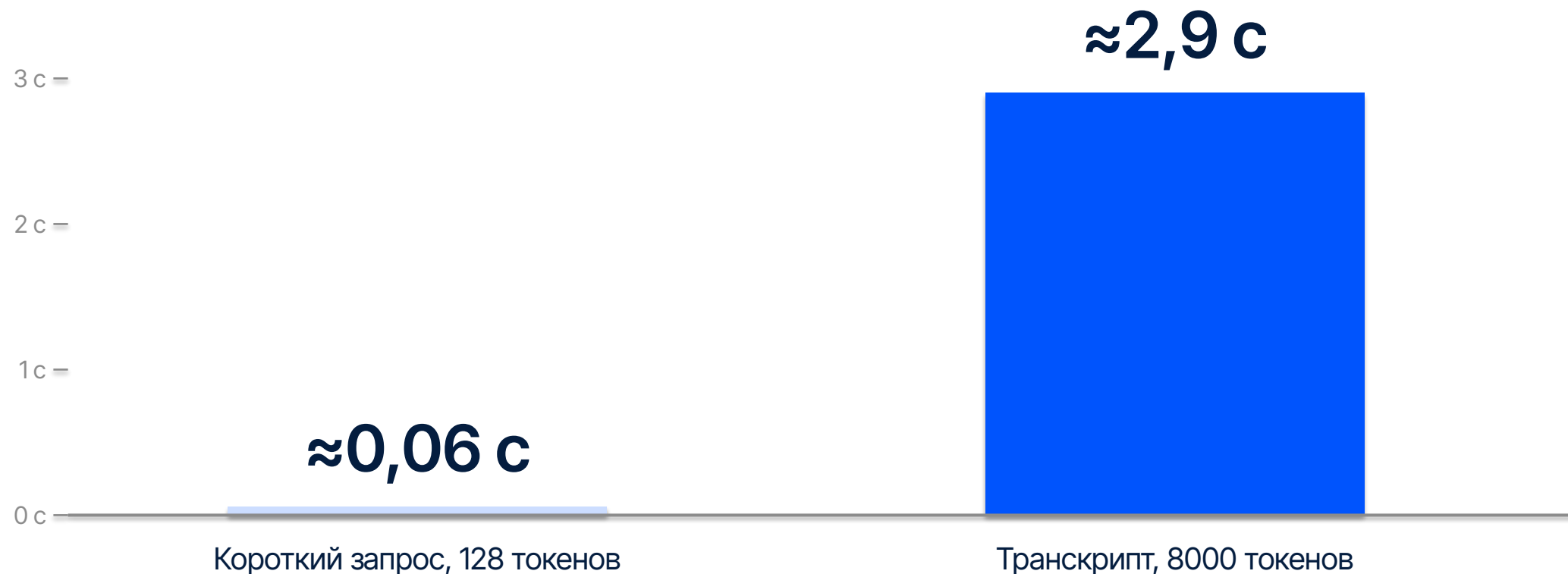
**45 tok/s**  
на небольших запросах

Раскатали



# Длинный вход меняет ожидание

Время до первого токена T4



Одиночные запросы · Генерация ответа ещё не началась

# Ожидание vs реальность



Ожидание

≈ 7 с

300 токенов при 45 tok/s



Реальность

бот

завис

а в логах те же 45 tok/s



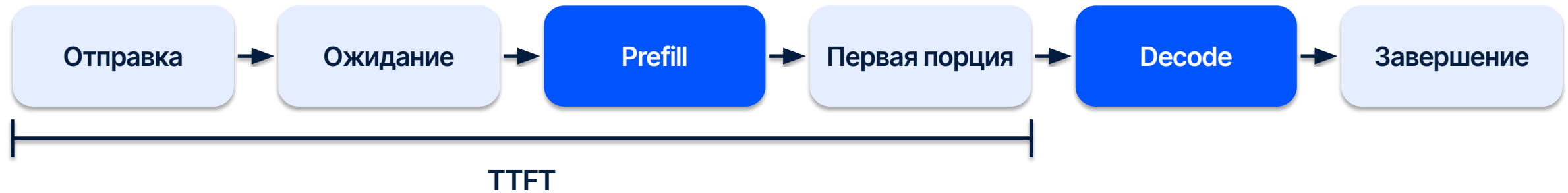
**Чего не учли:**

Длинные тексты на вход +  
одновременные обращения.

очередь + prefill = минуты  
до первого токена.

45 токенов в секунду описывали генерацию и молчали обо всём остальном

# Основные фазы



## Prefill

- читает весь промпт, строит KV cache
- обычно тяжёлый по вычислениям
- чем длиннее промпт - тем дольше
- **определяет TTFT - вместе с очередью**



## Decode

- генерирует по одному токenu
- на каждый token читает веса + KV cache
- обычно упирается в память
- **определяет TPOТ** - плавность генерации

# Бюджет prefill: 2048 и 16384

max-num-batched-tokens

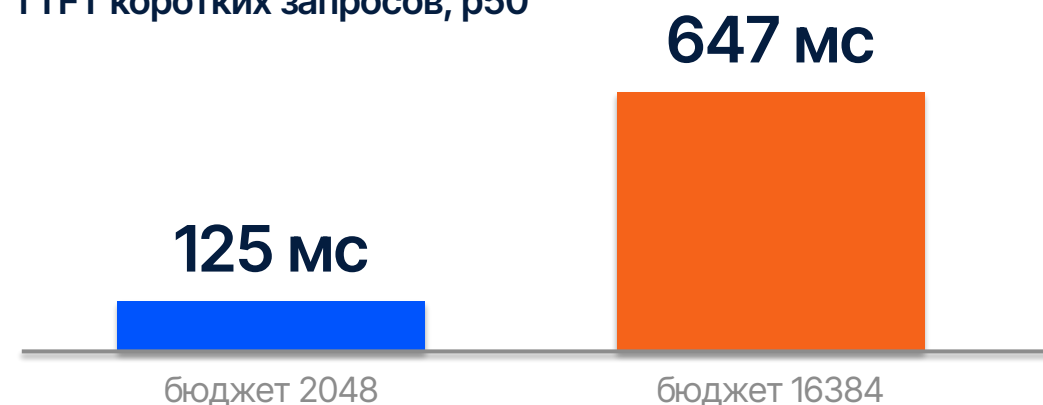
Смешанный тест на T4:

3 длинных запроса фоном + короткие

Без фона: TTFT короткого  $\approx 60$  мс

30 коротких проб; длинный фон: 24 запроса,  
 $c=3$

TTFT коротких запросов, p50



| Бюджет на шаг | TTFT long, p50 | TTFT short, p99 |
|---------------|----------------|-----------------|
| 2048          | 3201 мс        | 3881 мс         |
| 16384         | 9558 мс        | 7516 мс         |

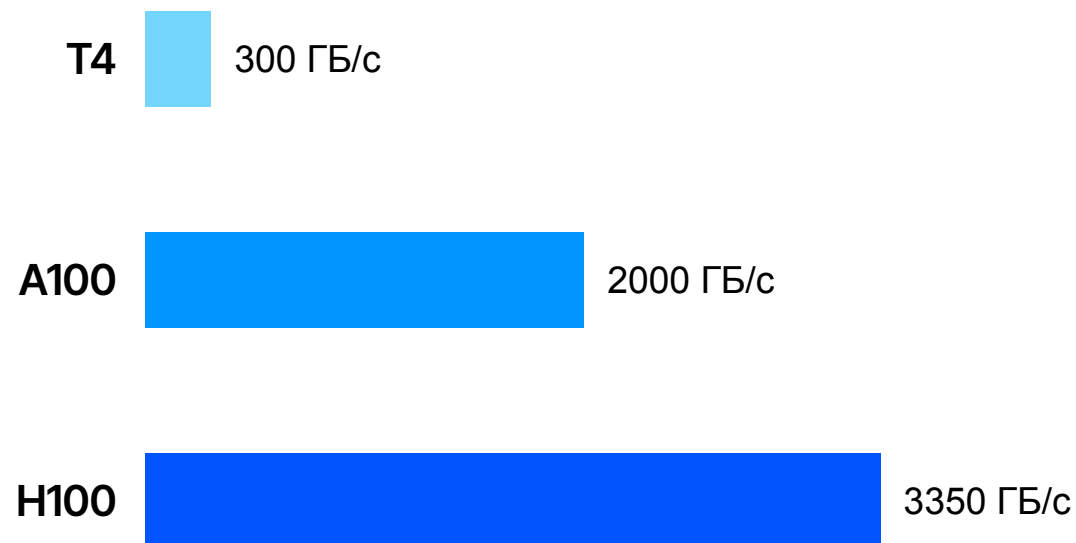
# Decode и пропускная способность памяти

T4: 300 GB/s ÷ ≈6 GB весов ≈ 50 чтений/с

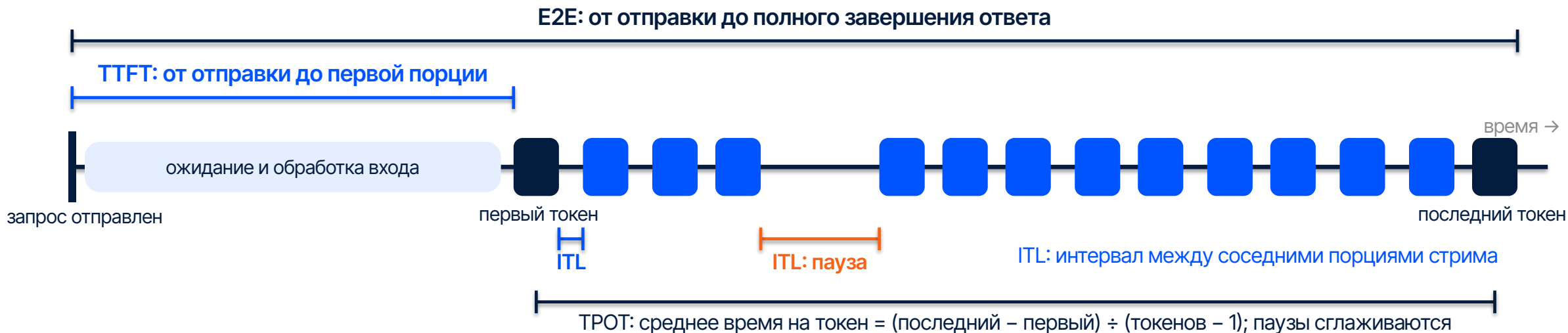
На реальной T4: 45,5 tok/s при c=1.

И 45,5 и 80 tok/s - скорость одного стрима.  
Запаса на четверых там нет

пропускная способность памяти, ГБ/с



# Основные метрики времени



## TTFT

сколько ждать  
первый токен

очередь + prefill +  
путь до клиента

## TPOT

среднее время на  
следующий токен

ровно ли бежит  
текст

## ITL

интервалы внутри  
стрима

паузы, которые  
прячет среднее

## E2E

полное время  
запроса

время готового  
результата

## Throughput

токены и запросы  
в секунду

сколько успели  
выдать



# Задержку описывают несколько метрик

Интерактивному чату важны TTFT и TROT,  
фоновым задачам - E2E и throughput.

Одной цифры не хватает ни тем, ни другим.



# Сценарий №2



Память длинного контекста



# RAG-ассистент

База знаний + история → в контекст модели

1

**A100 80 ГБ**

2

**Qwen3.6-35B-A3B FP8**

3

**max-model-len 200K**  
модель заявляет 262K токенов

# Из чего складывается расход VRAM

Веса + Память активных запросов + Overhead



## Веса модели

Все размещённые на карте параметры



## Память активных запросов

KV-кэш и другие данные, которые движок хранит для текущих запросов



## Overhead

Служебные расходы:  
Буферы, CUDA-графы, временные данные

МоЕ уменьшает активные вычисления, но все размещённые веса нужно хранить

# Квантизация весов



## Влияет на

- веса  $\div 2 \dots \div 4$  (FP8 / INT4)
- размещение в VRAM
- качество и скорость ответа



## Пример

- Qwen3.6-35B-A3B в нативном bf16 ~70 ГБ на A100 80GB впритык
- Qwen3.6-35B-A3B-FP8 уже ~35 ГБ

скорость зависит от GPU и ядер движка

# Квантизация KV cache



## Нужно помнить

- KV cache не становится FP8 или INT4 автоматически вместе с весами
- тоже влияет на качество ответа



## Пример

- Qwen3.6-35B-A3B-FP8 в config.json "dtype": "bfloat16"
- --kv-cache-dtype fp8 в prod.yaml; по умолчанию auto = dtype модели

**FP8 в имени модели не задаёт формат KV cache**

# Сколько KV приходится на токен

Qwen3.6-35B-A3B, данные из [config.json](#) модели

$$2 \times 10 \times 2 \times 256 \times 1 \text{ байт} = 10 \text{ KiB/tok}$$

К и V

full-attention слои

[layer\\_types](#)

KV-головы

[num\\_key\\_value\\_heads](#)

размер головы

[head\\_dim](#)

байты

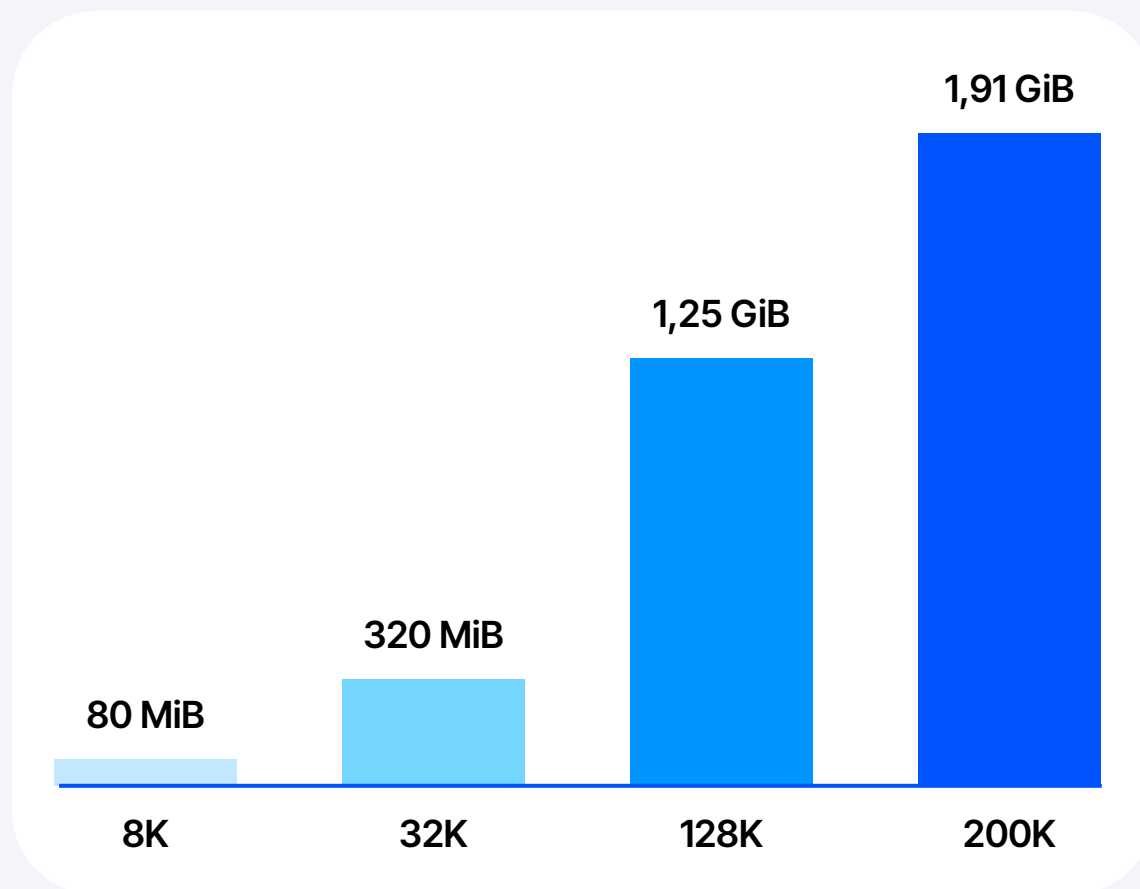
[kv-cache-dtype](#)

Сколько получится для 200 000 токенов?

# Контекст → память: один запрос

$10\text{KiB} \times 200\text{K} \approx 2\text{GiB}$

18 запросов  $\approx 36\text{ GiB}$



# Бюджет памяти одной A100

Ориентировочный расчёт для эксперимента

При запуске vLLM на A100 с `gpu-memory-utilization = 0.92`:

## 1762 полезных блока

результат расчёта доступной ёмкости при запуске Qwen3.6-35B-A3B-FP8 на A100

## 2096 токенов на блок

размер блока, выбранный движком для этой гибридной модели

Вес Qwen3.6-35B-A3B-FP8

$\approx 35 \text{ GiB}$

Память активных запросов

$1762 \times 2096 \times 10 \text{ KiB/tok} \approx 35,2 \text{ GiB}$

Служебная  
память

# От блоков к нагрузочному тесту



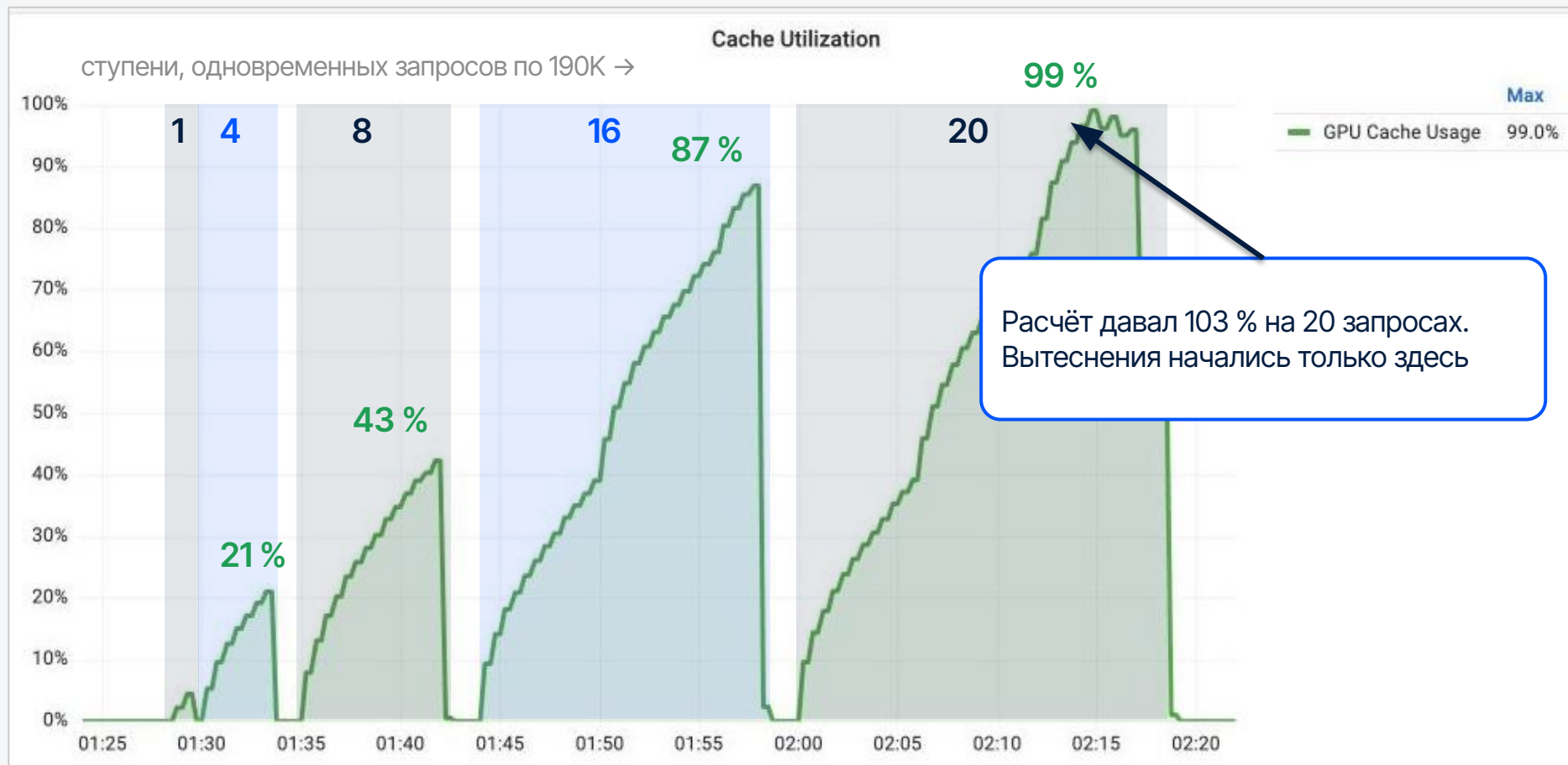
## Бюджет

- 1762 блока, 2096 токенов на блок
- 1 запрос 200K токенов  $\approx$  96 блоков

| Запросов в волне | Attention-блоков   | Заполнение  |
|------------------|--------------------|-------------|
| 4                | 364 / 1762         | 21%         |
| 8                | 728 / 1762         | 41%         |
| 16               | 1456 / 1762        | 83%         |
| <b>20</b>        | <b>1820 / 1762</b> | <b>103%</b> |

# Длинный контекст: заполнение кэша

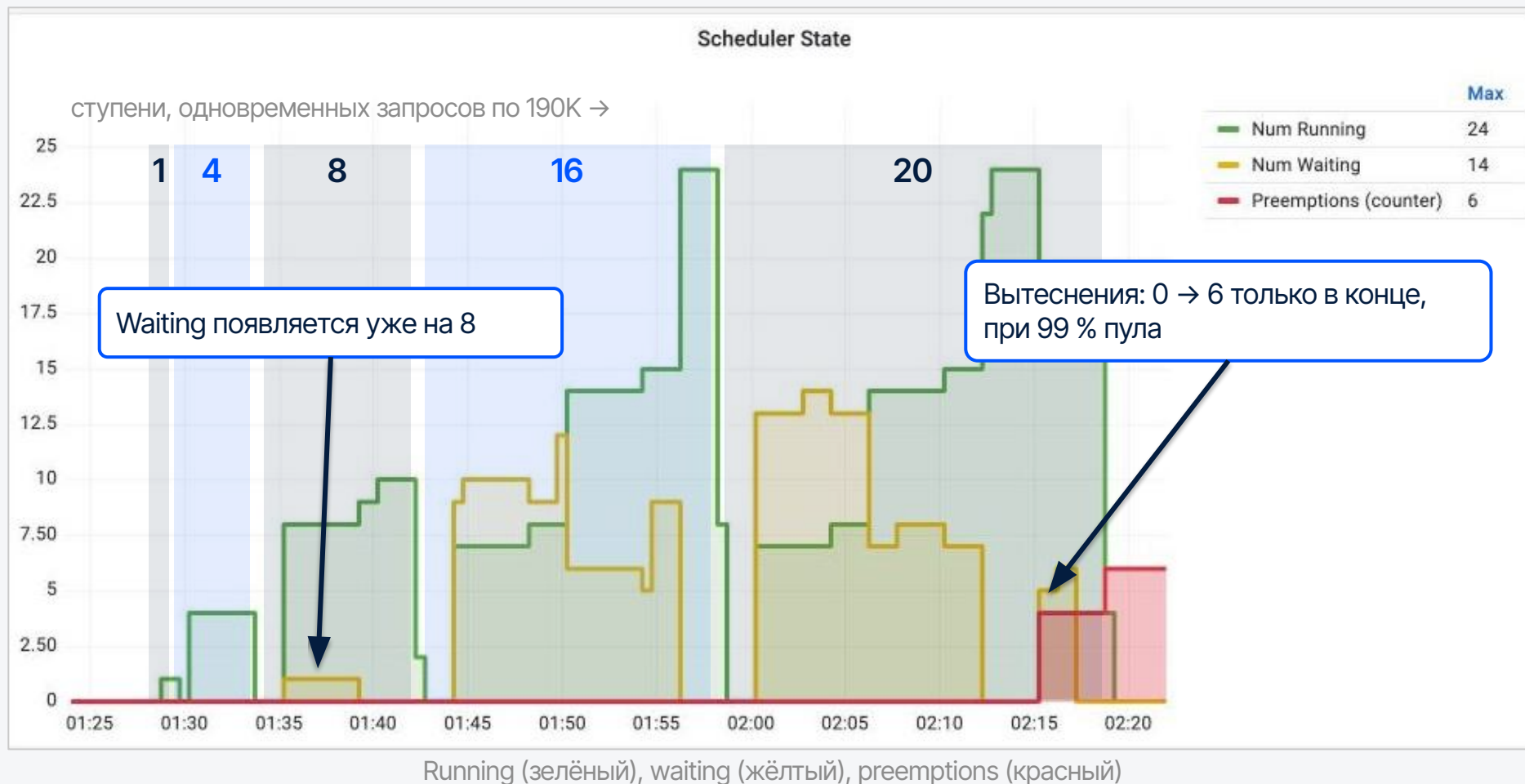
Qwen3.6-35B-A3B-FP8 · A100 · Волны: 1, 4, 8, 16, 20 запросов



Пики KV: 4,65% · 21,5% · 43,2% · 87,2% · 99,94% - ступени прогона

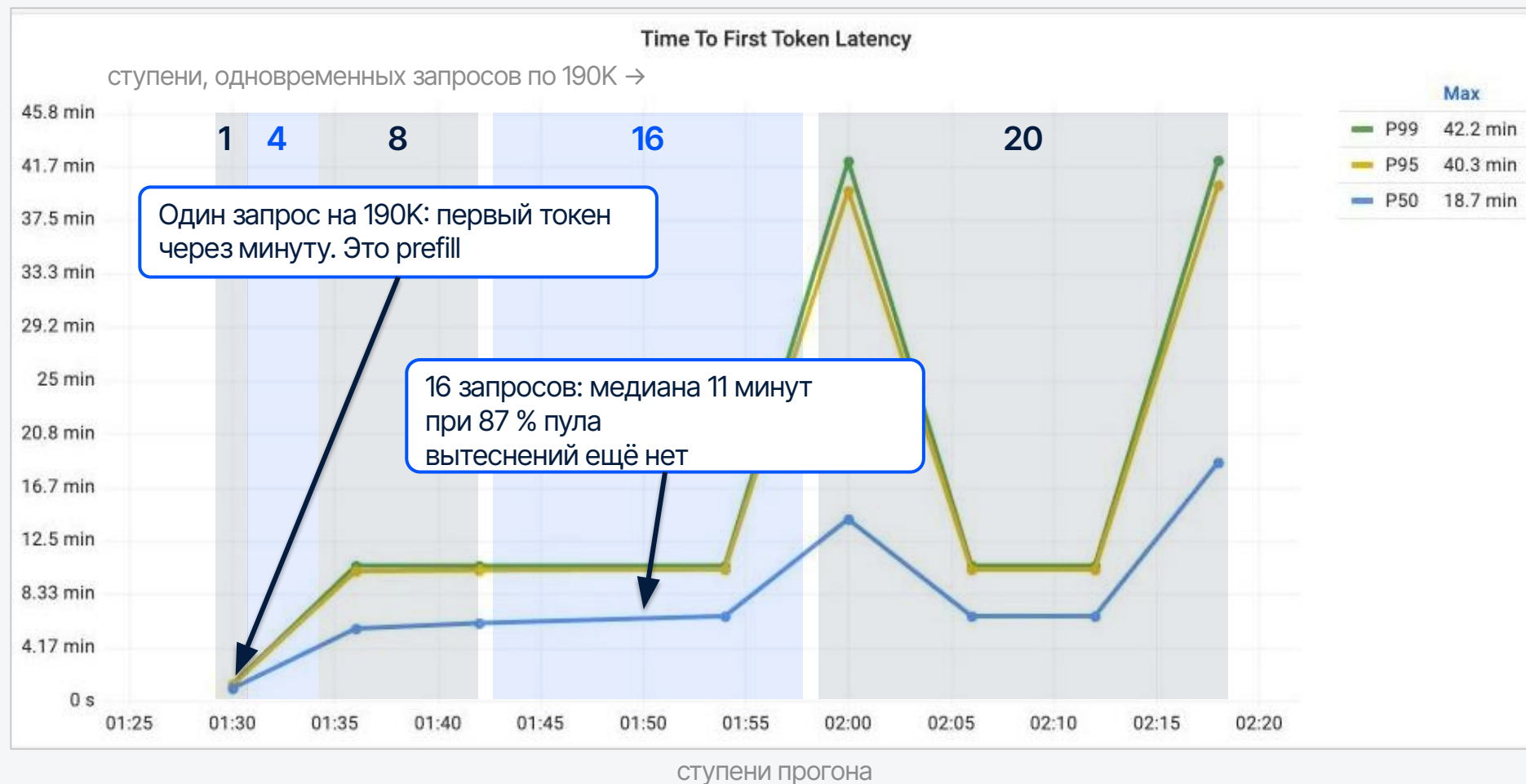
# Очередь и вытеснения при длинном входе

Та же Qwen3.6, тот же тест · Активных до 24 · Ожидающих до 14



# Задержка выросла до вытеснений

Та же Qwen3.6, тот же тест · Время до первого токена, минуты





# Длина контекста — это бюджет памяти

Прежде чем обещать продукту «мы поддерживаем длинные документы», посчитайте, сколько контекста вы реально сможете поддерживать при одновременных запросах.

**Свободная память ещё не гарантирует нужное время ответа**



# Сценарий №3



Поток запросов и требования сервиса



# Возвращаемся к задаче



## Ожидаемая нагрузка

60 пользователей

Один вопрос раз в 5 минут

Около 8000 токенов входа

До 400 токенов ответа

**В среднем 12 запросов/мин**



## Нагрузка дошла до расчётной

GPU загружена на 90 %

очередь пуста

Throughput растёт

**Часть пользователей жалуется -  
ожидание ответа до 20 с**

# GPU utilization показывает активность

~90%

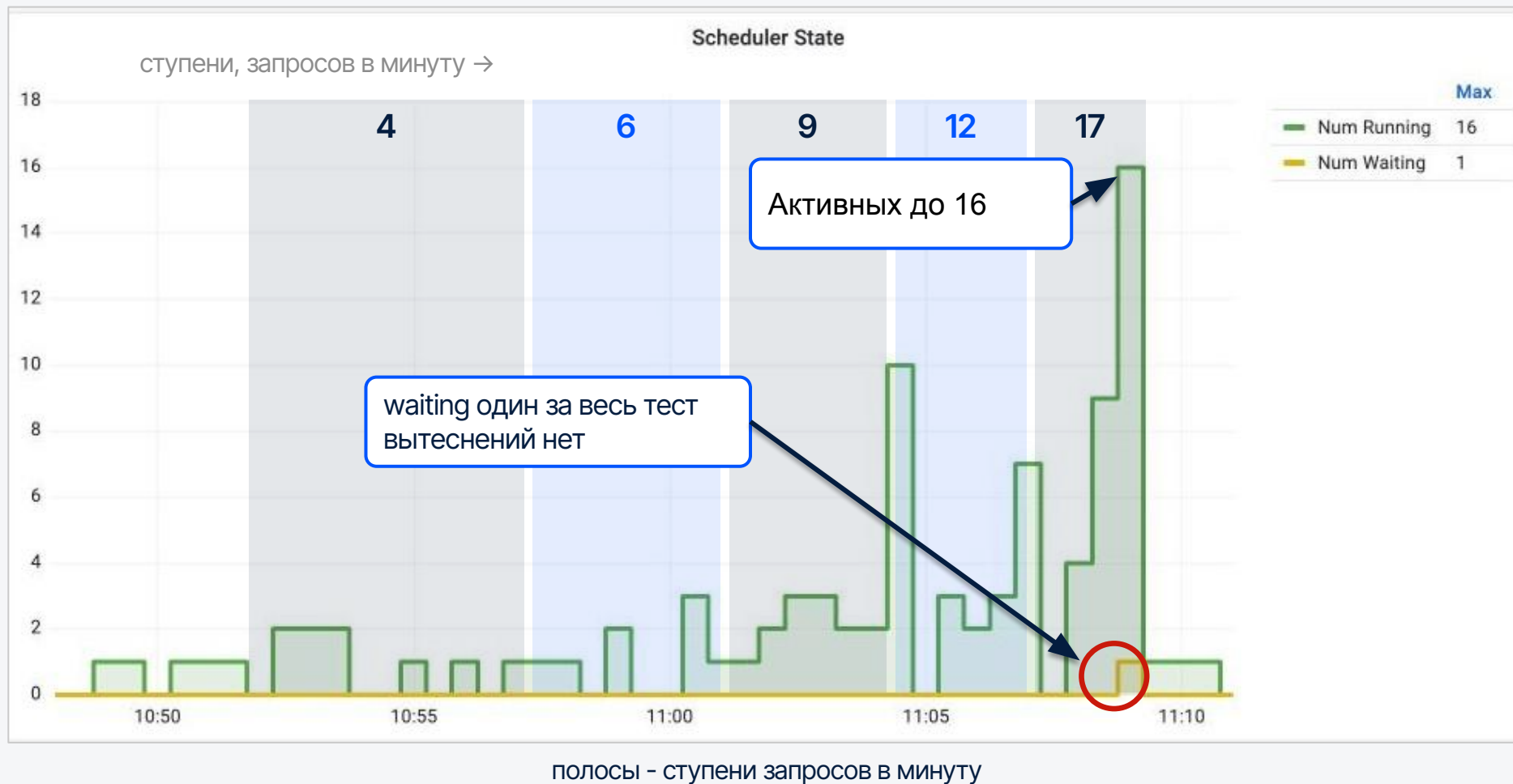
показывает лишь долю времени, когда на GPU  
хоть что-то выполняется

Доля времени, когда исполнялось хотя бы  
одно ядро GPU

- Не определяет долю запросов в SLO
- Для поиска ограничения нужны другие метрики  
и профилирование

# Пустая очередь не доказывает, что всё хорошо

Qwen3.8-27B-FP8 · Одна A100



# Средний TPOT скрывает паузы

Конкурентность 1

ITL p99

43 мс

Конкурентность 2

ITL p99

660 мс

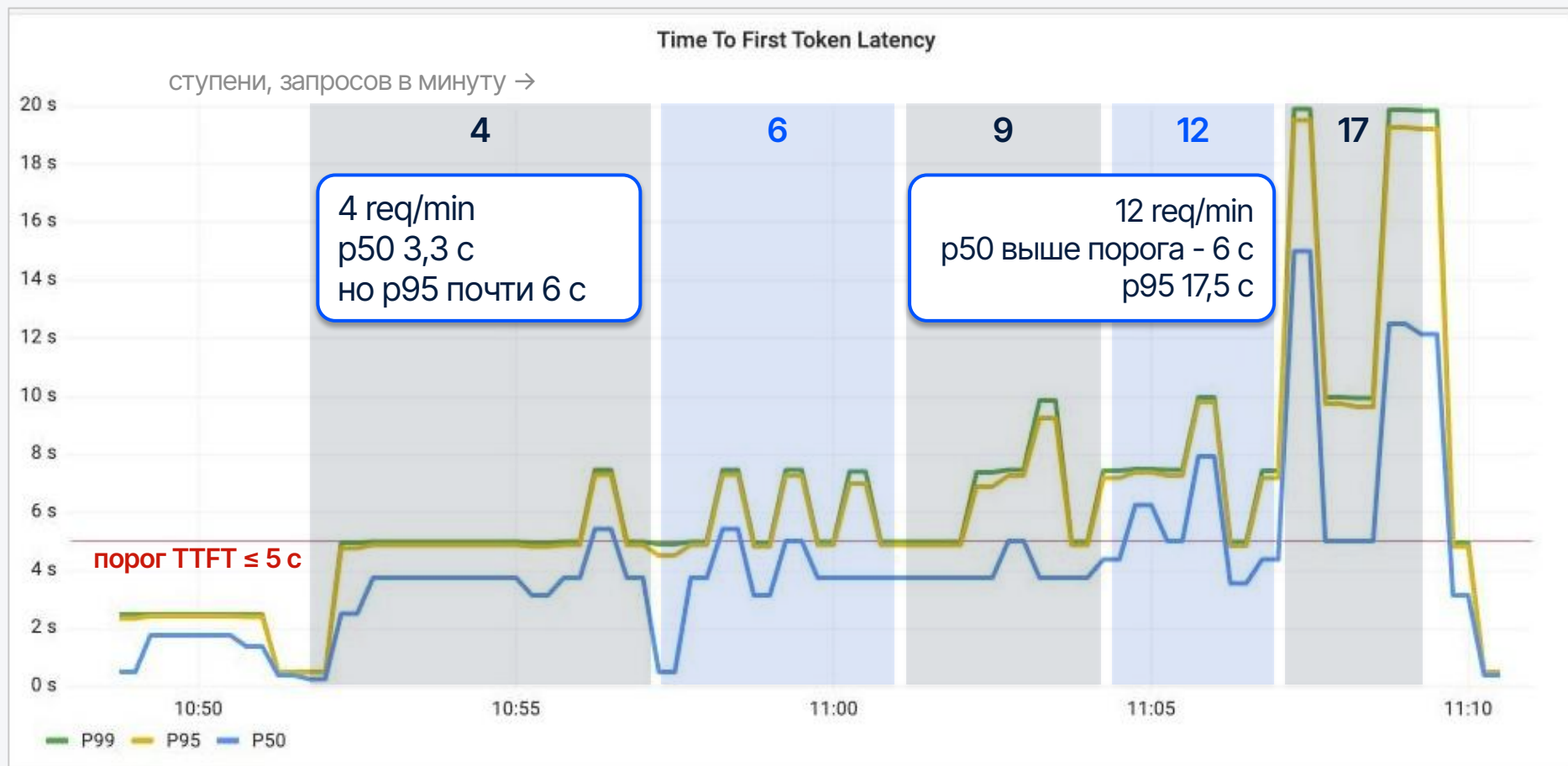
TPOT

13 мс

18 мс

# TTFT по ступеням нагрузки

Qwen3.8 · Одна A100



p50 / p95 / p99 · полосы - ступени подачи, запросов в минуту

# Goodput: считаем пользу, а не токены



## Throughput

Сколько запросов сервер обработал

6 запр/мин → 5,8

17 запр/мин → 15,6

Throughput - сколько сервер выдал



## Goodput

Сколько в рамках SLO

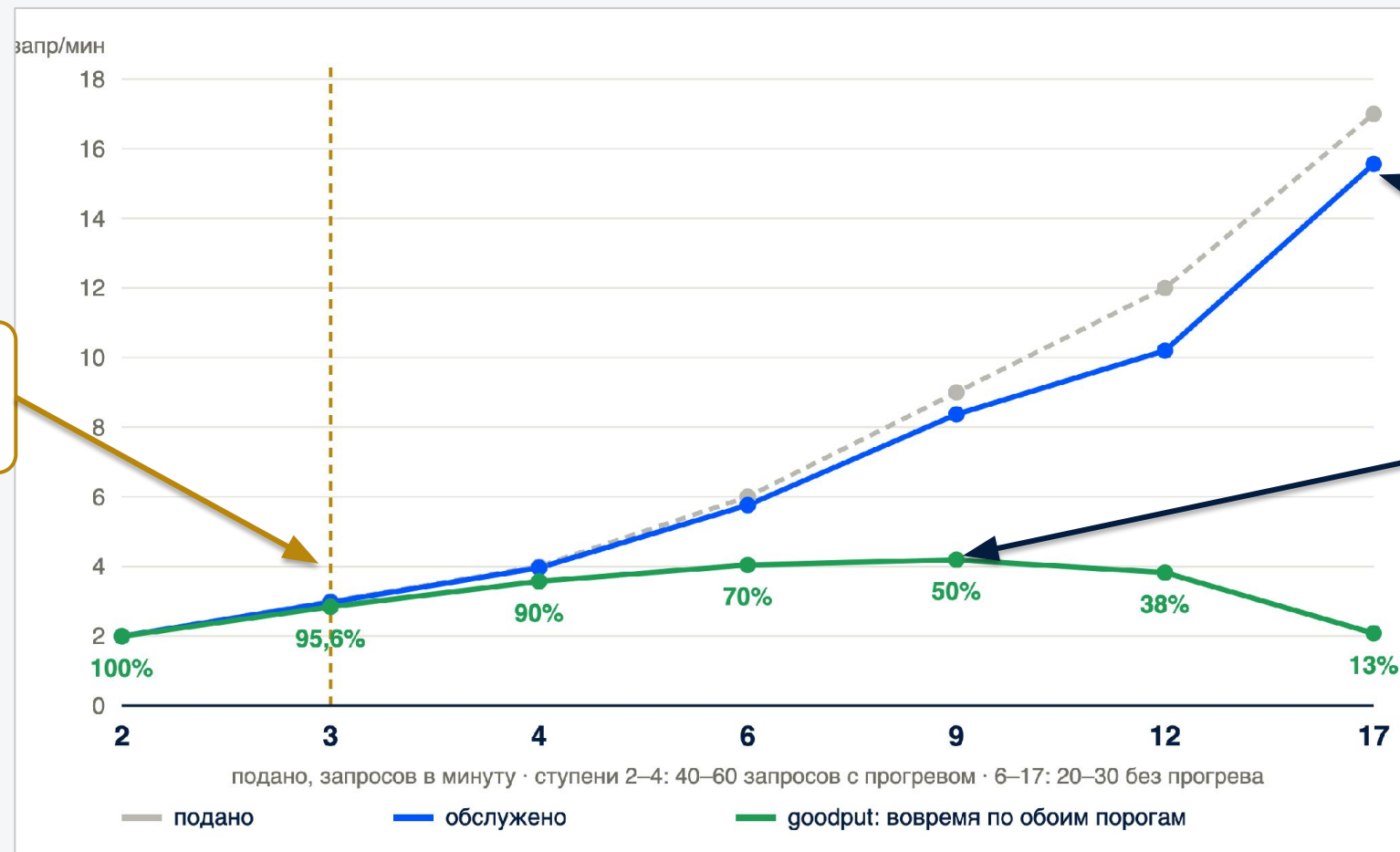
6 запр/мин → 4,0

17 запр/мин → 2,1

Goodput - сколько запросов получили вовремя

# Кривая нагрузки: где кончается goodput

Qwen3.8-27B-FP8 · Одна A100



~3 req/min  
рабочий максимум

При этом “обслужено”  
растёт до 15,6

Максимум goodput 4,2  
при 9 req/min



# Нагрузку проверяем вместе с требованиями SLO

Смотреть на перцентили - не только средние, goodput - не только throughput  
Ёмкость считать в терминах SLO



# Как повторять измерения





# Один прогон по шагам

1

**Паспорт прогона**  
что запущено и что  
проверяем

2

**Инструмент**  
как подаём и измеряем  
нагрузку

3

**Данные и профиль**  
что в запросах

4

**Результат**  
метрики и ограничения



# Паспорт прогона

## Раньше:

«Qwen 35B - 120 tok/s» в статье в Confluence

## Теперь:

Полный паспорт в git: revision, конфиг, команда  
и данные - прогон повторяется через месяц

## Модель и сервер

Qwen3.8-27B-FP8

A100-SXM 80 GB ×1

vLLM 0.27.1

kv fp8 · mm1 200000 · mns 64 · MTP 3

## Нагрузка

random 8192 in / 400 out

2 / 3 / 4 req/min

прогрев

## Результат

goodput 1,99 / 2,84 / 3,57

errors 0

## Ограничения

500 запросов

напрямую к реплике

# Инструменты

|                                 | vLLM bench | GuideLLM | GenAI-Perf | LLMPerf | Locust / k6 / JMeter |
|---------------------------------|------------|----------|------------|---------|----------------------|
| Измерять TTFT, TPOT и ITL       | ✓          | ✓        | ✓          | ✓       | ●                    |
| Goodput с порогами              | ✓          | ○        | ✓          | ○       | ●                    |
| Детали каждого запроса          | ✓          | ✓        | ✓          | ✓       | ●                    |
| Свои данные помимо синтетики    | ✓          | ✓        | ✓          | ○       | ✓                    |
| Прогрев перед замером           | ✓          | ✓        | ✓          | ○       | ●                    |
| Несколько частот одной командой | ○          | ✓        | ✓          | ○       | ✓                    |

✓ есть из коробки

● потребуется дополнительная настройка или свой код

# vLLM bench serve

- 0,05 запроса/с = 3 запроса/мин
- Пороги goodput задаются в миллисекундах

## Ключевые флаги нашего запуска

```
vllm bench serve ...  
  --request-rate 0.05  
  --burstiness 1  
  --num-prompts 45  
  --num-warmups 2  
  --goodput ttft:5000 tpot:50  
  --save-result --save-detailed
```

# Данные

## Синтетика

Синтетический вход нужной длины

## Реальные задачи

Сохраняем messages, sampling и распределения длин

```
--dataset-name random \  
  --random-input-len 8192 \  
  --random-output-len 400
```

```
--dataset-name custom \  
  --dataset-path prod_examples.jsonl
```

# Пять профилей нагрузки

| Профиль          | Что подаём              | Как проверяем           | Что смотрим                          |
|------------------|-------------------------|-------------------------|--------------------------------------|
| Короткий чат     | 512 входа / 256 выхода  | 1–8 одновременно        | Базовые задержки                     |
| Длинный промпт   | 16–64 тыс. на вход      | 1–4 одновременно        | Ожидание и влияние на другие запросы |
| Агентский цикл   | История, tools, префикс | Повторяем цепочку шагов | Рост контекста, кэш, качество        |
| Поток запросов   | Запросы разной длины    | Частота и всплески      | Доля в порогах и ошибки              |
| Память контекста | 32–128 тыс. на вход     | 1–8 одновременно        | KV, вытеснения и задержки            |

# Матрица контекст и конкурентность

Порог: p95 TTFT ≤ 2 с

4K - до восьми параллельных

8K - до двух

16K - только одиночные

32K - не проходит уже  
одиночный

|     | конкурентность → |        |        |        |
|-----|------------------|--------|--------|--------|
|     | 1                | 2      | 4      | 8      |
| 4K  | 313 мс           | 540 мс | 956 мс | 1,8 с  |
| 8K  | 658 мс           | 1,1 с  | 2,1 с  | 3,7 с  |
| 16K | 1,5 с            | 2,6 с  | 4,8 с  | 8,6 с  |
| 32K | 3,8 с            | 6,5 с  | 9,8 с  | 21,8 с |
| 64K | 10,6 с           | 18,3 с | 27,6 с | 61,7 с |

Qwen3.6-35B-A3B-FP8 · 1×A100-80 · порог 2 с

# Таблица резултата

| request rate, req/min | achieved, req/min | goodput, req/min | SLO attainment rate | TTFT p50 / p95, c |
|-----------------------|-------------------|------------------|---------------------|-------------------|
| 2                     | 1,99              | 1,99             | 100%                | 2,93 / 4,31       |
| 3                     | 2,97              | 2,84             | 95,6%               | 2,94 / 4,95       |
| 4                     | 3,96              | 3,57             | 90%                 | 3,10 / 5,77       |
| 12                    | 10,20             | 3,82             | 37,5%               | 6,00 / 17,46      |

Пороги TTFT  $\leq 5$  с, TPOT  $\leq 50$  мс, Ошибок 0

# Мониторинг: результат и диагностика



## Что видит пользователь

- Доля запросов, выполнивших оба порога
- TTFT, паузы и полное время ответа на пути пользователя
- Ошибки и таймауты



## Что в inference engine

- Активные и ожидающие запросы, время в очереди
- Заполнение KV-кэша и VRAM, вытеснения
- Доля попаданий в кэш

# Чеклист:

## Что проверить перед продам



- ☐ Конфигурация: GPU, модель, движок
- ☐ Запросы: данные, фактическая длина входа и ответа
- ☐ Нагрузка: темп, всплески, прогрев, кэш
- ☐ Результат: метрики, пороги, число запросов
- ☐ Надёжность: ошибки, длительность и повторные прогоны



# Теперь вы знаете, как проверить, сколько выдержит ваш сервис



## Спасибо! Вопросы?

Роман Песков · Циан