Санкт-Петербургский государственный университет

Кафедра системного программирования

Оптимизация алгоритмов injected queue в рантайме Tokio

Игорь Антонович Ерин, группа 21.Б10-мм

Научный руководитель: к.ф.-м.н. В. И. Гориховский, доцент кафедры системного программирования
Консультант: Р. С. Ефремов, «Эксперт по разработке ПО, ООО “Ядро Центр Программных Разработок”»

Рецензент: А.П. Черепанов, «Старший разработчик ПО, ООО “Ядро Центр Программных Разработок”»

Санкт-Петербург
2025

Введение

- `TATLIN.BACKUP`[1] — системы бекапов на `Rust`
- Rust — язык программирования общего назначения
  - `async` / `await` интерфейс
  - не фиксирует реализацию асинхронного рантайма
- `tokio`[2] — проект используемый в `TATLIN.BACKUP`
  - Рантаймы — `current_thread`, `multi_thread`
  - Стандартную библиотеку языка с асинхронным интерфейсом
  - Примитивы синхронизации
  - Коллекции
  - Таймеры
  - Профилировщики
- Популярная реализация планировщика
  - Go, Kotlin, tokio

[1] https://yadro.com/ru/tatlin/backup — официальный сайт TATLIN.BACKUP (Дата обращения: 4.1.2025)
[2] https://tokio.rs/ — официальный сайт проекта tokio (Дата обращения: 4.1.2025)

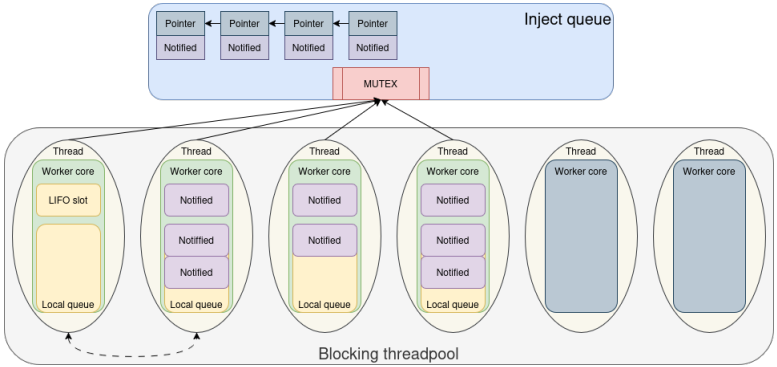
Постановка задачи

Целью работы является увеличение пропускной способности tokio в типичных для TATLIN.BACKUP сценариях.

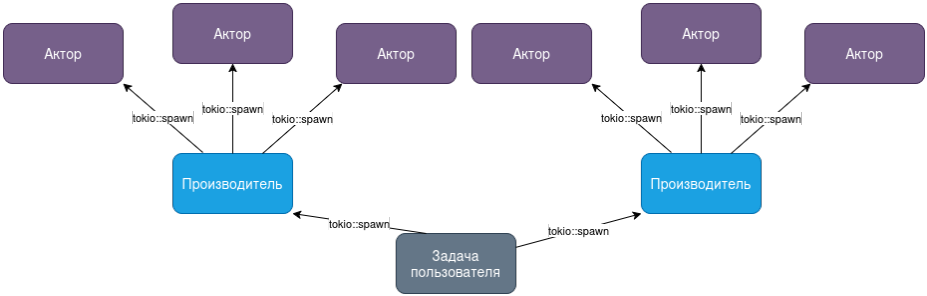
Задачи:

- Архитектурный анализ
- Разработка системы бенчмарков для оценки пропускной способности
- Анализ собранных метрик и локализация проблемы
- Проектирование решения, основанного на шардировании
- Реализация и анализ производительности решения

Многопоточный рантайм



Основной сценарий использования tokio



- Большинство задач ожидают асинхронные события
- Листовые задачи аллоцируются конкурентно

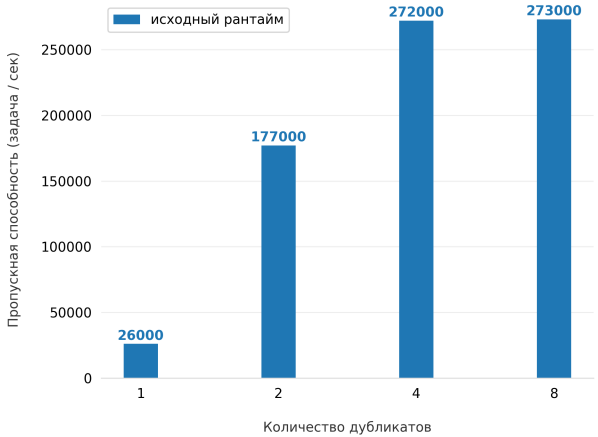
Система бенчмарков

- Недостатки бенчмарков в репозитории tokio
  - Атомарный подсчет выполненных задач
  - Использование API с высокой задержкой
- `tokiobench`[3] — проект с бенчмарками
  - YADRO VEGMAN Rx20 G2[4]
- Среднее отклонение пропускной способности не превысило 5 процентов
- Возможные решения
  - Шардирование состояние планировщика
  - Изменение реализации глобальной очереди

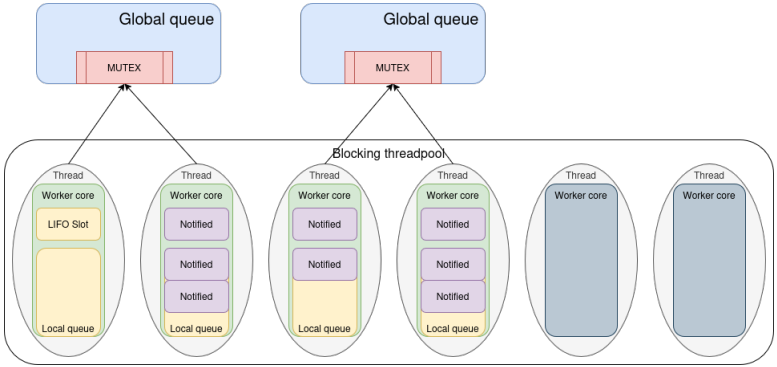
[3] https://github.com/IgorErin/tokiobench — репозиторий проекта tokiobench (Дата обращения: 19.4.2025)
[4] https://yadro.com/ru/vegman/rx20g2/specs — официальные характеристики (Дата обращения: 19.4.2025)

Предположение об использовании нескольких рантаймов

- С увеличением количества рантаймов растет пропускная способность
- Отсутствуют гарантии при использовании нескольких рантаймов
- Увеличивается когнитивная нагрузка при использовании нескольких объектов рантайма
- Придется переписать существующую систему с явным указанием необходимого рантайма



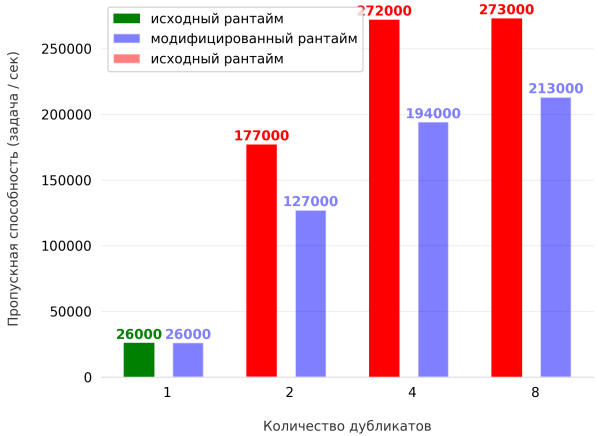
Предлагаемое решение



- Несколько глобальных очередей в рамках одного рантайма
- Изолированные группы воркеров

Пропусная способность решения

- Пропускная способность сравнима в случае одного исходного рантайма и рантайма с одной глобальной очередью
- Пропускная способность увеличивается с увеличением количества глобальных очередей в модифицированном рантайме



Заключение

- Проведен архитектурный анализ реализации, выявлены возможные уязвимости
  - Реализация глобальной очереди
  - Конкуренция исполнителей
- Создана система бенчмарков `tokiobench`[5]
- Реализовано[6] шардирование глобальной очереди
- Шардирование увеличило пропускную способность
  - В случае одного дубликата производительность не изменилась
  - Пропускная способность модифицированного рантайма меньше соотвествующего количества дубликатов исходных рантаймов в остальных случаях
- В TATLIN.BACKUP создана задача по улучшению работы с tokio

[5] https://github.com/IgorErin/tokiobench — проекта tokiobench (Дата обращения: 19.4.2025)

[6] https://github.com/IgorErin/tokio/pull/3 — реализация шардирования.
Имя пользователя: IgorErin (Дата обращения: 19.4.2025)