Содержание

Список сокращений и условных обозначений

- Термин1 -
- Термин2 -
- … -

Введение

На сегодняшний день в основе большинства разрабатываемых приложений: веб-серверов, кластеров обработки данных и.т.п. лежит инфрастуктурная основа в виде надёжной и производительной среды предоставления асинхронного исполнения - runtime. От него требуется обеспечение эффективной обработки задач, предоставления инструментов композиции и коммуникации между ними. Такая основа может быть встроена в сам язык, как например в Elixir и Go, или использоваться как отдельная библиотека - Kotlin Coroutines и Rust Tokio.

Любой такой инструмент строит свои абстракции исполнения поверх процессов или потоков, предоставляемых операционной системой. Поэтому неминуемо возникает потребность в синхронизации между ними. Синхронизация по своей природе может быть нескольких типов. Каждый из них предоставлет как преимущества, так и недостатки. Современные архитектуры имеют тенденцию распараллеливать вычислетельные процессы. Однако в большом числе случаев для синхронизации до сих пор используются инструменты крайне неэффектино масштабирующиеся вслед за архитектурой ЭВМ. В большинстве случаев - это блокирующая синхронизация, несмотря на то, что существуют способы производить операции в неблокирующем режиме, который открывает возможности к существенному масштабированию вычислений. Связано это с тем, что неблокиющая синхронизация довольно сложна в реализации, в отличие от блокирующей, а для комплексных структур данных эффективная реализация становится практически невозможной.

В рамках данной работы рассмотрена возможная примитива синхронизации в среде исполнения Tokio для языка Rust.

Введение в предметную область

Tokio + broadcast + pseudo …

Синхронизация по своей природе может быть блокирующей и неблокирующей. Первый тип блокирует прогресс всех исполняемых задач на время выполнения одной или выбранных нескольких задач. У такого типа синхронизации есть большое преимущество - он простой и не требует специального подхода к построения оперируемой структуры данных. Однако данный подход крайне неэффективно масштабируется. В худшем случая оперирование нескольких потоков или процессов в параллель может показать ипсполнение по производительности сравнительно худшее чем последовательное исполнение. Кроме того, существует проблема при которой поток или процесс захвативший блокировку будет временно снят с исполнения планировщиком задач операционной системы. В данном случае возникает полная остановка прогресса исполнения системы.

Основная часть

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri. (рисунок 1)

Рисунок 1 — пример изображения

1 Актуальность темы исследования

1.1 Актуальность темы исследования. Часть 1

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri.

1.2 Цель и Задачи

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri.

Заключение

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri.

Таблица 1 — Таблица

Таблица	**для**	**примера**
item1	$\sum_{k=0}^{n} k = 1 + ... + n$	description1
item2	$\sqrt{2}$	description2

Список использованных источников

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri.

Приложение

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri.