Эффективная реализация сопрограмм в управляемой среде исполнения

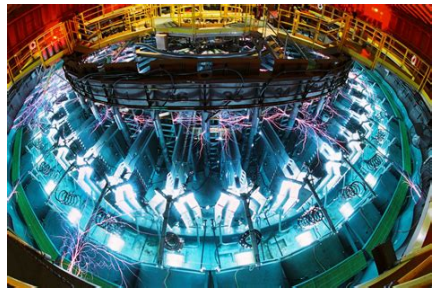
Евгений Пантелеев

Новосибирский государственный университет

Научный руководитель: Бульонков Михаил Алексеевич, канд. физ-мат наук
ИСИ СО РАН

Новосибирск
2021г.

(a) Серверы.



(b) Ускорители.

Существует множество задач, когда необходимо обрабатывать много независимых событий.

Сопрограммы

- **Сопрограмма** (с англ. coroutine) - программный модуль, организованный для обеспечения взаимодействия с другими модулями по принципу кооперативной многозадачности.
- Сопрограммы способны приостанавливать свое выполнение, сохраняя *контекст* (программный стек и регистры), и передавать управление другой.

Ключевые отличия от потоков ОС

Плюсы сопрограмм

- Переключение контекста сопрограммы требует меньше накладных расходов, чем потока.
- Как правило меньший размер стека, а значит, потребление памяти так же меньше.

Минусы

- Сопрограммы не способны исполняться параллельно.

Поддержка в языках программирования

(a) C++20.

(b) C#.

(c) Go.

Project Loom

Fibers and Continuations

- Project Loom – проект на базе OpenJDK, целью которого является разработка сопрограмм для языка Java.
- На данный момент уже доступна ранняя версия проекта.

Цели и задачи.

Цель: реализация прототипа сопрограмм в Java.
Поставленные задачи:

- Разработать тесты для сравнения эффективности различных реализаций корутин.
- Реализовать переключение контекста сопрограмм.
- Поддержать сборку мусора(????).
- Сравнить производительности реализаций.

Работа проводится на базе Huawei JDK.

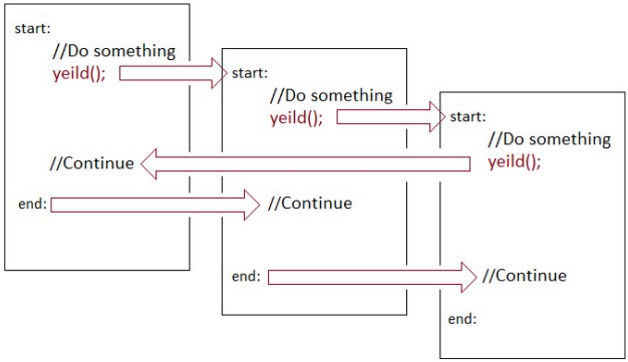
Тесты производительности

Тесты создавались для измерения 2 параметров.

- Скорость переключения контекста.
- Потребление памяти.

Репозиторий с тестами: https://github.com/minium2/coroutines-benchmark

Переключение контекста.



Подходы к реализации:

- OpenJDK: копирование стека сопрограммы при переключении.
- HuaweiJDK : изменение указателя стека.

Сборка мусора

Что-то про сборку мусора

Результаты

- Создан набор тестов производительности для языков Kotlin, Go, Java(”Loom Project”).
- Реализовано переключение контекста сопрограмм.
- Реализована трассировка ссылок объектов на стеках сопрограмм(???).
- Получены результаты тестов производительности.

Результаты: скорости переключения

Ubuntu, x64, 31 Гб ОЗУ
Каждое значение усреднено по 100 измерениям.

Сопрограмм, шт.	Число переключений, 1/сек.	
	HuaweiJDK	OpenJDK
100	483’656 (-/+ 4’113)	1’900’009 (-/+ 19’732)
1000	476’467 (-/+ 4’517)	1’775’239 (-/+ 20’491)
5000	435’239 (-/+ 2’942)	1’703’631 (-/+ 30’498)
10000	419’376 (-/+ 4’266)	1’924’971 (-/+ 234982)
50000	372’378 (-/+ 2’719)	1’518’349 (-/+ 152899)

Результаты: потребление памяти

Ubuntu, x64, 31 Гб ОЗУ

Число сопрограмм, шт.	Размер физической памяти, 1/сек.		
	HuaweiJDK	OpenJDK	Go
100	34M	130M	3040K
1000	38M	161M	3105K
5000	59M	187M	3156K
10000	85M	193M	3308K
50000	287M	202M	3407K

Применение сопрограмм.

- Реализация бесконечных списков, итераторов, генераторов.
- Написание асинхронного и неблокирующего кода (та причина, из-за которой сопрограммы стали востребованы сейчас)

План дальнейших работ

- Переделать функцию переключения контекста.
- Реализация возможности миграции сопрограмм с одного потока на другой.
- Синхронизация: поддержка synchronized блоков.
- Переключение сопрограммы при вызове ввода вывода.

Спасибо за внимание!