УДК 004

Обзор использования Goroutines языка программирования Go в целях ускорения работ программного обеспечения

Ленкин Алексей Викторович
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
Цель исследования рассмотреть использование Goroutines языка программирования Go, описать его, показать практическое применение. Методами исследования являются анализ теоретической информации об объекте исследования, а также проведение эксперимента практического использования Goroutines. Итогом статьи является выделение особенностей использования Goroutines и его преимущества над аналогами.
Ключевые слова: goroutines, go, многопоточность, потоки

An Overview of Using Goroutines Go Programming Language to Speed Software

Lenkin Aleksei Viktorovich
Sholom-Aleichem Priamursky State University
Student

Abstract
The purpose of the study is to consider the use of the Goroutines Go programming language, describe it, and show practical application. Research methods are the analysis of theoretical information about the object of study, as well as the experiment of practical use of Goroutines. The result of the article is to highlight the features of using Goroutines and its advantages over analogues.
Keywords: goroutines, go, multithreading, threads

Сфера разработки программного обеспечения для ЭВМ последнее время развивается очень быстро. Вместе с этим развивается и аппаратная часть у ЭВМ, но они до сих пор не имеют безграничных ресурсов. Для того чтобы оптимизировать свои программы в такой ограниченной среде большинство разработчиков прибегает к использованию в них принципов параллелизма или многопоточности.

Большинство языков программирования (ЯП) не поддерживает использования многопоточности либо её использование является очень сложным и не стоит затраченных на её подключение и внедрение времени. Одним из языков, где подключение является легким это Go, в частности функции типа Goroutines.

Цель исследования рассмотреть использование Goroutines языка программирования Go, описать его, показать практическое применение.

Исследованиями в данной теме занимались следующие авторы. Денискин А.В. показал использования «Многопоточности в языке программирования C#» [1]. «Многопоточность и параллелизм в Unix подобных ОС на платформе x86» была продемонстрирована Булыгиным Г.А. и Козловой Л.П. [2]. Трифанов В.Ю. описал статью на тему «Обнаружение состояний гонки в Java-программах на основе синхронизационных контрактов» [3]. Бакулев А.В., Бакулева М.А., Козлов М.А. и Скворцов С.В. показали «Технологии разработки параллельных программ для современных многоядерных процессоров» [4].

Goroutines - это функции или методы, которые запускаются одновременно с другими функциями или методами. Goroutines можно рассматривать как легкие потоки. Стоимость создания Goroutines крошечная по сравнению с обычными потокам. Следовательно, это позволяет приложениям Go работать одновременно с тысячами Goroutine одновременно [5].

Работают они по следующему алгоритму. Когда запускается новый Goroutine, вызов другой Goroutine немедленно возвращается. В отличие от функций, элемент управления не ожидает завершения работы Goroutine. Элемент управления немедленно переходит к следующей строке кода после вызова Goroutine, и любые возвращаемые значения из Goroutine игнорируются.

Опишем, как именно Goroutines отличаются от потоков:

1. Использование памяти. Goroutines чрезвычайно дешевы на память по сравнению с потоками. Размер стека составляет всего несколько килобайт, и размер стека может увеличиваться и уменьшаться в зависимости от потребностей приложения, тогда как в случае потоков размер стека должен быть указан и фиксирован.
2. Goroutines мультиплексируются в меньшее количество потоков. Там может быть только один поток в программе с тысячами Goroutines. Если какая-либо программа в этом потоке блокирует ожидание ввода пользователя, то создается другой поток и остальные программы перемещаются в новый поток. Все выполняется автоматически во время исполнения, и мы, как программисты, абстрагируемся от этих сложных деталей и получаем чистый API для работы с параллелизмом.
3. Горутины общаются, используя каналы. Каналы по своей конструкции предотвращают возникновение условий гонки при доступе к общей памяти с помощью Goroutines. Каналы можно рассматривать как трубопровод, по которому общаются Goroutines.

Такая технология значительно упрощает разработку программ любого уровня, использующих принципы параллелизма.

Продемонстрируем работу Goroutines на простом примере. Для этого напишем программу, которая будет считать возведение чисел от 0 до 9 в степень 100000. Без использования Goroutines эта операция выполнялась бы линейно и заняла много времени, но с ними все числа будут возводиться в степень одновременно и результат будет получен моментально (рис.1).

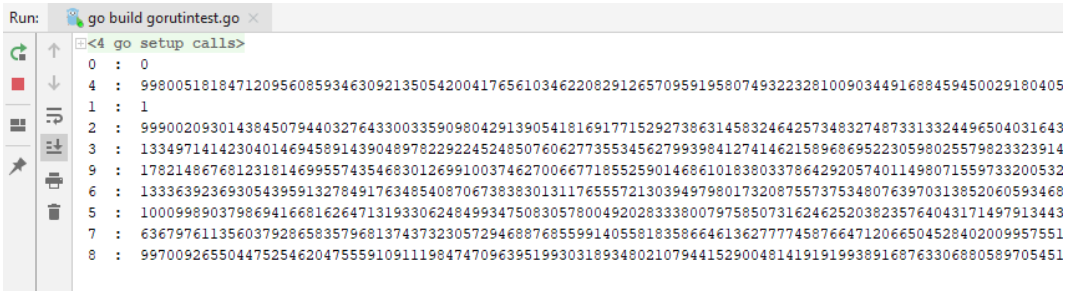

Рисунок 1. Результат выполнения программы с использованием Goroutines

Листинг программы представлен ниже:

```
package main

import (
    "fmt"
    "math/big"
)

func test(n int64) {
      fmt.Println(n, " : ", new(big.Int).Exp(big.NewInt(n),
big.NewInt(100000), nil))
}
func main() {
    for i := 0; i < 10; i++ {
       go test(int64(i))
    }
    var input string
    fmt.Scanln(&input)
}
```

Таким образом, подход к использованию Goroutines вместо обычных потоков является перспективным выбором для разработчиков программного обеспечения с поддержкой параллелизма, так как намного легче подключается и занимает в разы меньше памяти, что значительно упрощает разработку.

Библиографический список

1. Денискин А.В. Многопоточность в языке программирования C# // Academy. 2017. № 2 (17). С. 21-24.
2. Булыгин Г.А., Козлова Л.П. Многопоточность и параллелизм в Unix подобных ОС на платформе x86 // В сборнике: Актуальные проблемы инфотелекоммуникаций в науке и образовании (АПИНО 2018) VII Международная научно-техническая и научно-методическая конференция. Сборник научных статей. В 4-х томах. Под редакцией С.В. Бачевского. 2018. С. 93-96.
3. Трифанов В.Ю. Обнаружение состояний гонки в Java-программах на основе синхронизационных контрактов // Компьютерные инструменты в

образовании. 2012. № 4. С. 16-29.
4. Бакулев А.В., Бакулева М.А., Козлов М.А., Скворцов С.В. Технологии разработки параллельных программ для современных многоядерных процессоров // Экономика, статистика и информатика. Вестник УМО. 2014. № 6. С. 211-215.
5. Goroutines - Concurrency in Golang // golangbot.com URL: https://golangbot.com/goroutines/ (дата обращения: 04.09.2019)
6. Многопоточность - Введение в программирование на Go // Введение в программирование на Go URL: http://golang-book.ru/chapter-10-concurrency.html (дата обращения: 04.09.2019).