|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ Информатики и систем управления

КАФЕДРА Теоретической информатики и компьютерных технологий

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1**

**по курсу: «Разработка параллельных и распределенных программ»**

Студент: Яровикова А. С.

Группа: ИУ9-41Б

Преподаватель: Царев А.С..

*.*

*2022 г.*

# Условие задачи

Лабораторная работа №1: распараллеливание алгоритма вычисления произведения двух матриц.

1. Две квадратные матрицы A и B размерности n сначала перемножить стандартным алгоритмом: 𝑐𝑖𝑗 = для получения матрицы С той же размерности. Замерить время вычисления, сравнить с временем при вычислении элементов матрицы С не по строкам, а по столбцам.
2. Размер матриц подобрать таким образом, чтобы время выполнения на машине было не слишком непоказательно малым (меньше нескольких минут), но и не чересчур большим (несколько часов). Использовать библиотечные функции для вычисления произведений матриц нельзя.
3. Затем конечную матрицу С условно разделить на примерно равные прямоугольные подматрицы и распараллелить программу таким образом, чтобы каждый поток занимался вычислением своей подматрицы. Матрицы А и В для этого разделить на примерно равные группы строк и столбцов соответственно.
4. Сделать для разного количества потоков (разных разбиений), также замерить время вычисления, сравнить с вычислениями стандартным алгоритмом.
5. Также по окончании вычислений сравнивать получившуюся матрицу с той, что была вычислена стандартным алгоритмом, для проверки правильности вычислений (проверка во время выполнения задачи не входит)

# 2. Характеристики устройства

1. Процессор Intel(R) Core(TM) i5-4310U CPU @ 2.00GHz, 2601 МГц, ядер: 2, логических процессоров: 4. 8 ГБ ОЗУ.
2. Язык golang (1.17.6), для распараллеливания вычислений используются потоки и горутины.
3. Утилита time, для вычисления времени потраченного на вычисление произведения матриц.

**3. Листинг текста программы**

 package main  
  
import (  
 "bufio"  
 "fmt"  
 "log"  
 "math/rand"  
 "os"  
 "reflect"  
 "strconv"  
 "strings"  
 "sync"  
 "time"  
)  
  
var res [][]int  
  
func randomFill(n int) [][]int {  
 matrix := make([][]int, n)  
 for i := 0; i < n; i++ {  
 matrix[i] = make([]int, n)  
 for j := 0; j < n; j++ {  
 matrix[i][j] = rand.Intn(10)  
 }  
 }  
 return matrix  
}  
  
func standardMatrixMul(a [][]int, b [][]int) [][]int {  
 n := len(a)  
 c := make([][]int, len(a))  
 for i := 0; i < n; i++ {  
 c[i] = make([]int, n)  
 for j := 0; j < n; j++ {  
 sum := 0  
 for k := 0; k < n; k++ {  
 sum += a[i][k] \* b[k][j]  
 }  
 c[i][j] = sum  
 }  
 }  
 return c  
}  
  
func columnsMatrixMul(a [][]int, b [][]int) [][]int {  
 n := len(a)  
 c := make([][]int, len(a))  
 for i := 0; i < n; i++ {  
 c[i] = make([]int, n)  
 }  
 for j := 0; j < n; j++ {  
 for i := 0; i < n; i++ {  
 sum := 0  
 for k := 0; k < n; k++ {  
 sum += a[i][k] \* b[k][j]  
 }  
 c[i][j] = sum  
 }  
 }  
 return c  
}  
  
func parallelCalcMatrix(a [][]int, b [][]int, res [][]int, s int, e int, n int, wg \*sync.WaitGroup) {  
 defer wg.Done()  
 for i := s; i < e; i++ {  
 res[i] = make([]int, n)  
 }  
 for i := s; i < e; i++ {  
 for j := 0; j < n; j++ {  
 sum := 0  
 for k := 0; k < n; k++ {  
 sum += a[i][k] \* b[k][j]  
 }  
 res[i][j] = sum  
 }  
 }  
}  
  
func areMatricesEqual(a [][]int, b [][]int) bool {  
 for i, v := range a {  
 if reflect.DeepEqual(v, b[i]) {  
 continue  
 } else {  
 return false  
 }  
 }  
 return true  
}  
  
func main() {  
 var c [][]int  
 file, err := os.Open("test1.txt")  
 if err != nil {  
 log.Fatal(err)  
 }  
 defer file.Close()  
 scanner := bufio.NewScanner(file)  
 scanner.Scan()  
 sc := scanner.Text()  
 n, err := strconv.Atoi(sc)  
 if err != nil {  
 log.Fatal(err)  
 }  
  
 a := make([][]int, n)  
 for i := range a {  
 a[i] = make([]int, n)  
 }  
 for j := 0; j < n; j++ {  
 scanner.Scan()  
 sc = scanner.Text()  
 words := strings.Split(sc, " ")  
 for i := 0; i < n; i++ {  
 a[j][i], err = strconv.Atoi(words[i])  
 if err != nil {  
 log.Fatal(err)  
 }  
 }  
 }  
 scanner.Scan()  
 b := make([][]int, n)  
 for i := range b {  
 b[i] = make([]int, n)  
 }  
 for j := 0; j < n; j++ {  
 scanner.Scan()  
 sc = scanner.Text()  
 words := strings.Split(sc, " ")  
 for i := 0; i < n; i++ {  
 b[j][i], err = strconv.Atoi(words[i])  
 if err != nil {  
 log.Fatal(err)  
 }  
 }  
 }  
 fmt.Println("matrix A")  
 for i := 0; i < len(a); i++ {  
 fmt.Println(a[i])  
 }  
 fmt.Println("matrix B")  
 for i := 0; i < len(b); i++ {  
 fmt.Println(b[i])  
 }  
 start1 := time.Now()  
 c = standardMatrixMul(a, b)  
 end1 := time.Now()  
 fmt.Println("matrix AxB")  
 for i := 0; i < len(c); i++ {  
 fmt.Println(c[i])  
 }  
 dim := len(c)  
 start2 := time.Now()  
 c = columnsMatrixMul(a, b)  
 end2 := time.Now()  
 fmt.Println("matrix AxB another type")  
 for i := 0; i < len(c); i++ {  
 fmt.Println(c[i])  
 }  
 fmt.Println("============")  
 var th int  
 fmt.Println("Input num of threads")  
 fmt.Scan(&th)  
 /\* for th threads\*/  
 s := 0  
 d := n / th  
 e := 0  
 res = make([][]int, dim)  
 var wg sync.WaitGroup  
 start3 := time.Now()  
 for k := th; k > 0; k-- {  
 wg.Add(1)  
 if k == 1 {  
 e = n  
 } else {  
 e += d  
 }  
 go parallelCalcMatrix(a, b, res, s, e, dim, &wg)  
 s = e  
 }  
 wg.Wait()  
 end3 := time.Now()  
 fmt.Println(res)  
 fmt.Println("standard matrix multiplication takes", end1.Sub(start1))  
 fmt.Println("matrix multiplication through columns takes", end2.Sub(start2))  
 fmt.Println("parallel matrix multiplication through threads takes:", end3.Sub(start3))  
 fmt.Println("threads", th)  
 fmt.Println(areMatricesEqual(c, res))  
}

**4. Время работы**

|  |  |  |  |
| --- | --- | --- | --- |
| Количество потоков | Размерность матрицы | | |
| 1000х1000 | 1500х1500 | 2000х2000 |
| 1 | 13.8 | 46 | 106 |
| 2 | 11 | 36.5 | 94.6 |
| 4 | 8.6 | 32.7 | 91 |
| 8 | 8.5 | 31.8 | 81 |
| 16 | 8.4 | 31.7 | 56 |
| 32 | 8.2 | 30 | 55 |

