ГУАП

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ст. преподаватель |  |  |  | П. А. Степанов |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №7 |
| Исследование производительности программного продукта с помощью профайлера |
| по курсу: УПРАВЛЕНИЕ КАЧЕСТВОМ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4631 |  |  |  | С.А. Гришин |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2018

**Задание**

В рамках лабораторной работы необходимо произвести функциональное тестирование Rich интерфейса приложения

Задание на лабораторную работу.

10. Взять задание из лабораторной работы номер 2. Модифицировать полученный код чтобы разработанную функцию можно было выполнять большое количество раз (например, 10000).

11. Подключиться к исполняемому коду профилировщиком (или использовать встроенный в IDE) и проанализировать – какой фрагмент кода занял больше всего процессорного времени.

12. Предложить способы оптимизации

**Вариант 9: Функция, производящая поиск заданного набора строк в текстовом файле.**

Поиск должен уметь находить любую строку из набора, при этом должен правильно обрабатывать переносы текста.

Использовать алгоритм: Ахо-Корасик.

**Решение**

Результаты перового запуска:

goos: windows

BenchmarkOfw-4 2000000 899 ns/op 56 B/op 3 allocs/op

PASS

30ms 30ms 50:func (m \*Matcher) Match(s string) []int {

. . 51: curNode := m.root

10ms 160ms 52: m.resetMark()

. . 53: var p \*trieNode = nil

. . 54:

. 50ms 55: ret := make([]int, 0)

. . 56:

150ms 150ms 57: for \_, v := range s {

230ms 2.11s 58: for curNode.child[v] == nil && curNode != m.root {

. . 59: curNode = curNode.fail

. . 60: }

110ms 850ms 61: curNode = curNode.child[v]

40ms 40ms 62: if curNode == nil {

. . 63: curNode = m.root

. . 64: }

. . 65:

40ms 40ms 66: p = curNode

720ms 720ms 67: for p != m.root && p.count > 0 && !m.mark[p.index] {

. . 68: m.mark[p.index] = true

30ms 30ms 69: for i := 0; i < p.count; i++ {

90ms 430ms 70: ret = append(ret, p.index)

. . 71: }

. . 72: p = p.fail

. . 73: }

. . 74: }

. . 75:

В данном случае время тратится на поиск в map. Можно убрать map и сделать вместо него массив. Мы пожертвуем китайскими языками и другими, но получит ускорение выполнения.

Результаты после оптимизации:

goos: windows

BenchmarkOfw-4 10000000 184 ns/op 16 B/op 2 allocs/op

PASS