МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

профессионального образования

**«Дальневосточный федеральный университет»**

**ШКОЛА ЕСТЕСТВЕННЫХ НАУК**

**Кафедра информационных систем управления**

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 5**

По дисциплине «Методы, средства и технологии информационных систем управления»

На тему «Самоорганизующиеся таблицы»

|  |
| --- |
| Выполнил студент гр. Б8319 |
| О.В. Константинов |
|  |
| Проверил ст. преп. |
| Е.А. Елсукова |
|  |
| (зачтено/не зачтено) |

г. Владивосток

2017

# Аннотация

Данный отчет подготовлен в рамках задания по предмету «Методы, средства и технологии информационных систем управления». Отчет призван систематизировать знания о способах работы с саморегулирующимися таблицами. Структура документа разделена, в соответствии с требованием преподавателя, на несколько разделов: «Постановка задачи», «Функциональное описание», «Примеры выполнения программы», «Вывод по работе».

# Постановка задачи

Требуется разработать демонстрационную программу, реализующую работу со структурой поиска, которая содержит в качестве ключей – интервал, а значения – число.

В качестве структуры данных будет выступать список элементов:

|  |  |  |  |
| --- | --- | --- | --- |
| Ключ | Начало интервала | Конец интервала | Порядковый номер |

**Ключ** – ключ поиска (значение).

**Начало интервала, Конец интервала** – промежуток значений, при попадании в который будет выбран данный ключ.

**Порядковый номер** – место данного элемента в списке, упорядоченном по убыванию вероятности (т.е. в идеально организованном списке).

Результат работы программы должен представлять собой график зависимости степени оптимальности организации данных от количества элементов поиска для каждого способа балансировки таблицы (перемещения в начало и транспозиции).

Количество элементов поиска: n/2, n, 2\*n, 5\*n, 10\*n. Где n – количество ключей в списке.

# Функциональное описание

Программа представляет собой оконное приложение для ОС Windows NT (не ниже 5.1). Приложение было реализовано в IDE Visual Studio версии 2015, на языке C#.

Структурно, проект состоит из одной файла: Project.cs.

В программе, данные представляют собой список объектов класса TableCell:

class TableCell

{

public int Key; // Ключ поиска

public int StartInterval; // Начало интервала выбора ключа

public int EndInterval; // Конец интервала выбора ключа

public int SerialNumber; // Порядковый номер – место данного элемента в идеальном с.

}

Сам же список этих элементов содержится в классе Table:

class Table

{

public enum BalancingType

{

MovingToTheBeginning, // Метод перемещения в начало

Transposition // Метод транспозиции

};

private List<TableCell> Cells; // Список ячеек

public int IntervalWidth; // Ширина диапазона ключей (max(EndInterval))

public BalancingType Balance; // Политика оптимизации таблицы

}

Перемещение данных происходит при каждом обращении к элементу из Table в зависимости от политики оптимизации таблицы:

public TableCell this[int key]

{

get

{

for (int i = 0; i < Cells.Count(); i++)

if (Cells[i].StartInterval <= key && key <= Cells[i].EndInterval)

{

var res = Cells[i];

switch (Balance)

{

case BalancingType.MovingToTheBeginning:

Cells.Remove(res);

Cells.Insert(0, res);

break;

case BalancingType.Transposition:

if (i > 0)

Swap(Cells, i - 1, i);

break;

default:

throw new NotImplementedException();

}

return res;

}

return null;

}

}

Сам же результат балансировки считается после каждого обращения к Table при помощи метода int DegreeOfBalance():

public int DegreeOfBalance()

{

int result = 0;

for (int i = 0; i < Cells.Count(); i++)

result += Math.Abs(Cells[i].SerialNumber - i);

return result;

}

# Примеры выполнения программы

ВХОД (ПАРАМЕТРЫ ЭКСПЕРЕМЕНТА):

const int count = 25; // Количество ключей

const int range = 100; // Диапазон генерируемых чисел на каждом шаге

// Последовательность количества ключей для поиска

int[] countOfKeys = { count / 2, count, 2 \* count, 5 \* count, 10 \* count };

// Последовательность количества ключей для формирования отчета (не используется)

int[] countForReport = Enumerable.Range(1, 50).Select(x => x \* count / 5).ToArray();

ВЫХОД (ТАБЛИЦА):

Moving to the beginning method

Balance: 136 (for 12 appeals)

Balance: 158 (for 25 appeals)

Balance: 110 (for 50 appeals)

Balance: 108 (for 125 appeals)

Balance: 128 (for 250 appeals)

Transposition method

Balance: 178 (for 12 appeals)

Balance: 168 (for 25 appeals)

Balance: 158 (for 50 appeals)

Balance: 154 (for 125 appeals)

Balance: 120 (for 250 appeals)

ВХОД (ГРАФИК):

# Вывод по работе

В результате проведенной работы была достигнута поставленная цель: изучение основных методов работы с саморегулирующимися таблицами. Можно сделать вывод, что при высокочастотных обращениях к саморегулирующейся таблице метод транспозиции дает наилучший результат. А метод перемещения в начало применим при необходимости получении быстрого выигрыша при малых количествах запроса к таблице.