Searching with data structures

Gerben Aalvanger
Studentnumber: 3987051
Utrecht University

Erik Visser
Studentnumber: 3470806
Utrecht University

William Kos
Studentnumber: 3933083
Utrecht University

Sam van der Wal
Studentnumber: 3962652
Utrecht University

18 februari 2015
Teacher: Peter de Waal

1 Introduction

Searching values and storing them in memory is a key concept in computer science. For optimal speed and complexity, data is stored in a lot of different data structures, for example a tree or a skiplist. Every data structure has its own advantages, some perform better on insertion, while other perform better on finding a specific value. In this document we will describe how we will compare data structures and actually compare them. We will start by proposing a research question and some sub-questions. Secondly we specify the problem we want to research and the scope of our project. After that we explain our experiments in terms of criteria, test data and scenarios. And last we will show in different ways our test results and compare them to eachother and end with a conclusion.

2 Onderzoeksbeschrijving afgeleid van onderzoeksplan

3 Beschrijving van de experimenten

We will seperate the experiment in several section:

3.1 Build

In the build we have "build" the different datastructures. For the build operation we have build from15 different unsorted datasets, this is divided in datasets of size 10.000, 100.00 and 1.000.000, each size occuring 5 times. For each of these 15 options we have also tested the sorted and reverse sorted set. Each dataset is tested 30 times. This means that we have conducted all tests with 1350 distinct datasets.

3.2 Search

We have searched in the pre-build datastructures of section 3.1. The amount of searches is half the amount of elements in the datastructure. All the values of the searches are determined random and are garanteed to be in the datastructure. Like in the build section we have performed all search-tests 30 times.

We wanted to know if finding the minimum or maximum through the getMin (see section 3.5) and getMax (see section 3.6) methods of the datastructures are faster then searching for the minimum or maximum value with this search method. This test is conducted 1000 times on the datasets described in section 3.1

3.3 Insert

We have constructed new datastructures from scratch by repeatedly inserting new elements. Like in section 3.1 we have tested the insert actions for 15 sets of elements (set sizes: $10.000, 100.000, 1.000.000$) and each of these tests have been performed 30 times.

3.4 Delete

We have performed delete operations on the pre-build datastructures of section 3.1. The amount of deletes is half the amount of elements in the datastructure. All the values of the deletes are determined random and are garanteed to be in the datastructure. Like in the build section we have performed all delete-tests 30 times.

3.5 getMin

We performed getMin operations on the pre-build datastructures of section 3.1. The amount of getMin operations in a test is 1000 since it returns always the same value. GetMin returns the lowest value in the datastructures. Like in the build section we performed all getMin-tests 30 times.

3.6 getMax

We performed getMax operations on the pre-build datastructures of section 3.1. The amount of getMax operations in a test is 1000 since it returns always the same value. GetMax returns the highest value in the datastructures. Like in the build section we performed all getMax-tests 30 times.

3.7 extractMin

We performed extractMin operations on the pre-build datastructures of section 3.1. The amount of extractMin operations in a test is half the amount of the elements in the datastructure. ExtractMin returns the lowest value in the datastructures and then deletes this entry. Like in the build section we performed all extractMin-tests 30 times.

3.8 extractMax

We performed extractMax operations on the pre-build datastructures of section 3.1. The amount of extractMax operations in a test is half the amount of the elements in the datastructure. ExtractMax returns the highest value in the datastructures and then deletes this entry. Like in the build section we performed all extractMax-tests 30 times.

4 Weergave van eindresultaten

4.1 Tabellen

4.2 Grafieken

4.3 Toelichting

4.4 Hypotheses en uitwerking statistische tests

Welke veronderstellingen gaan we testen. Dit is een verdere uitwerking van de onderzoeksvraag. Minimaal 4 echt verschillende , meer mag ook. Denk aan: Vergelijkingen: Parameterinstelling 1 levert lagere looptijd parameterinstelling 2 Relaties: Er is een correlatie tussen het aantal klanten dat pizza's besteld en de looptijd van het algoritme Statistische test moeten worden uitgevoerd met Excel

5 Discussie en conclusie

6 Reflectie

6.1 In hoeverre heb je de onderzoeksvraag beantwoord?

6.2 Heb je je onderzoeksplan kunnen uitvoeren of heb je het bijgesteld? Zo ja, hoe en waarom?

6.3 Tegen welke moeilijkheden ben je aangelopen in het projetc?