A SYSTEM TO SUGGEST IDEAL DATA STRUCTURES FOR JAVA PROGRAMS

A PROPOSAL PRESENTED BY

R.N.G.J.H. NAWARATHNA
(S/16/417)

to the Department of Statistics and Computer Science of the

FACULTY OF SCIENCE

in partially fulfillment of the requirement

for the award of the degree of

B.Sc. (Honours) in Computer Science

of the

UNIVERSITY OF PERADENIYA

SRI LANKA

2021

TABLE OF CONTENTS

LIST OF TABLES

CHAPTER 1
INTRODUCTION

With the tech development in recent years, computer hardware and software have become more emerged within one another. As it continues to develop, the thing that matters is the performance of those applications with the relevant hardware. Performance is one primary concern when it comes to the quality of software. For example, in a business application, it directly affects the revenue of that business. The Quality of Service (QoS) provided by these enterprise applications can be identified as performance, availability, and security. The proposed system here is mainly focused on the performance of these application programs.

Over the years, Java has become one of the major languages for developing enterprise applications. Java has helped developers to create large scalable applications more securely. The reason developers choose Java is that it is more secure, reliable, and powerful enough to create such complex enterprise applications. People often talk about the performance of these applications and how they can be improved in terms of performance. Performance models to predict performances and suggest necessary changes in the application are reliable ways of improving software quality. Also, this helps the software engineering industry bring higher efficiency and better quality to the software that they build.

1.1 Problem Statement

In software, data structures are a significant concern because they help to organize and store data so that developers can perform operations on data more efficiently. These are used in almost every application in various types such as Arrays, Maps, Linked Lists, Stacks, etc. Data structures play a huge responsibility in terms of the performance of so-called applications. So, when it comes to developing software, developers use multiple data structures. There can be situations that they might think at a particular point that the data structure used is good; however, it is not afterward. The only way to find out this is by measuring the performance of the software.

Developers use various software testing mechanisms to ensure their applications perform well in the consumer's hand. However, often, these testing mechanisms are not ideal in most cases.

Building an exact system to perform testing for a real-world application is hard to achieve since it requires too much money and time. Sometimes it is not easy to test software like this because reproducing certain functionalities is challenging and time-consuming.

Performance modeling comes in handy to help developers predict their applications' performance and improve the quality. Performance models provide performance predictions rather than performance observations. These performance predictions are beneficial to identify performance bottlenecks and to timely anticipate future performance problems.

As a vastly grown platform, Java uses many advanced data structures a lot and helps developers create more complex enterprise applications for consumers. Often, performance may vary from one data structure to another in an application depending on the use case. Developers do not usually measure performance when the application runs from time to time because it is time-consuming. So, what the proposed system does is that it calculated the performance of a particular Java program when it gets executed and analyzes those performances. After that, it suggests what data structure can be used instead to increase the performance immediately. It relies on the performance of the particular program and hence more accurate the prediction is.

1.2 Objectives

- Propose a performance modeling system for Java.
- Propose a system to analyze the performance of Java programs and suggest better data structures to use instead.
- To increase the efficiency and reliability of Java programming in terms of software engineering.

CHAPTER 2

LITERATURE REVIEW

Over the years, software engineering has gradually been developed to a point where it can be divided into many subcategories. Software Performance Engineering (SPE) is indeed a significant category that falls under software engineering. As it is defined, Software Performance Engineering (SPE) represents the entire collection of software engineering activities and related analyzes used throughout the software development cycle, which are directed to meeting performance requirements[1]. The performance of the software is one of the main factors deciding the product's success.

Resources are required to keep a system running within the parameters of the non-functional performance requirements. The performance of a system is decided by how the system interacts with the resources, and we can define resources as hardware(CPU, bus, I/O and storage, network), logical resources(buffers, locks, semaphores), processing resources(processes, threads)[1]. As the paper[1] describes it, a determining factor for performance is that resources have a limited capacity, so they can potentially halt/delay the execution of competing users by denying permission to proceed. There are several SPE activities; Identify concerns, Define and analyze requirements, Predict Performance, Performance testing, Maintenance and evolution, and Total system analysis[1].

Several types of research have been done related to performance modeling over the past years. According to Wu and Woodside[2], component-based software engineering brings efficiency and quality to software development using reusable and configurable software components. Hence, it helps to increase the quality of performance engineering. Also, they suggest that performance should be based on the pre-defined software component. So, the same approach can be used to build a performance model for Java applications as well. According to research named Performance Monitoring of Java Applications[3], the performance monitoring system should monitor the following elements of execution behaviour.

- The invocation of methods

- Object allocation and release.
- Thread creation and destruction.
- Mutual exclusion and cooperation between threads.

The results should include attributes that can be used to calculate performance measures[3]. These data might be stored in a database and accessed through an application programming interface(API). When it comes to monitoring, there are two types: event-driven and time-driven. Time-driven monitors at certain time intervals, whereas event-driven observes events in a system[3].

In terms of Java performance monitoring, there are a few tools that we can use. Visual VM, JVM, JMeter, and Sentry are those tools that fall under mentioned tools. We can use these tools to measure the performances of various Java applications and hence get an initial idea about the performance of a particular program. Other than these, Java Profiler is a tool that monitors Java bytecode constructs and operations at the JVM level. This tool is also important because just a running code is not enough for production applications. So, programmers need to look at memory allocations and so on to see how the product works. For that, Java Profiler is a good tool. The mentioned tools Java Visual VM falls into this category.

Java is a program that uses data structures a lot to get the work done. Measuring the overall performance of an application includes data structures that have been used in that application as well. A data structure is that performance tasks efficiently organize, process, retrieve, and store data. Depending on the approach for the program, the data structure that needs to be used varies. Hence the application's performance varies as well.

In most cases, sorting algorithms are used inside the implantation of the data structure. The choice of sorting algorithm depends on various parameters like the amount of memory or machine time needed for running a program, how fast and accurately it sorts a list[6]. So, considering all these things, a unique system can be proposed to measure the performances of Java programs that use data structures and suggest various data structures depending on the usage inside the application.

CHAPTER 3

METHODOLOGY

3.1 Languages

- Java programming language along with Java Runtime Environment(JRE) is used to implement the relevant applications.
- Python programming language is used to achieve the machine learning approaches.

3.2 Tools

- Visual VM
- JMeter
- Sentry
- Java VM

3.3 Focused Data Structures

- Arrays
- Linked Lists
- Stacks
- Queues
- Hash Tables
- Trees
- Heaps
- Graphs

PROGRESSION TIMELINE

Table 01: The Progression Timeline and the evaluation of the project.

Duration / Action	June	July	August	September	October	November	December	January	February
Proposal and Requirement Analysis									
Design									
Implementation									
Documentation									
Testing and Debugging									

REFERENCES

1. Woodside, M., Franks, G., Petriu, D.C.: The future of software performance engineering. In: Future of Software Engineering (FOSE) (2007)
2. Wu, X., Woodside, M.: Performance modeling from software components. SIGSOFT Softw. Eng. Notes 29(1), pp. 290–301 (2004)
3. M. Harkema, D. Quaetel, B.M.M. Gijsen, R.D. van der Mel: Performance Monitoring of Java Applications (2002)
4. Wenlong Li, Eric Li, Ran Meng, Tao Wang, Carole Dulong: Performance Analysis of Java Concurrent Programming: A Case Study of Video Mining System (2006)
5. Andreas Brunnert, Christian Vogele, Helmut Krcmar: Automatic Performance Model Generation for Java Enterprise Edition (EE) Applications (2013)
6. Sourabh Shastri: Studies on the Comparative Analysis and Performance Prediction of Sorting Algorithms in Data Structures (2014)