Comparative Analysis of Multi-Threaded and Non-Threaded

Implementations of the Merge Sort Algorithm

(Proposal)

By

Bernice Templeman

Florida Institute of Technology

CIS 5898 – Projects in Computer Information Systems

For

Dr. Stephen Johnson

September 9, 2016

Introduction 3

Sorting Decisions 3

Java Threads 3

BTMS Proposition 3

Algorithms / Project Solution 4

Merge-Sort ALgorithm 4

Threads 4

Data Set Size 4

Timing 4

Implementation 5

Platform 5

Development Tools 5

Programming Language and Code Libraries 5

Implementation Procedure 5

Schedule 6

References 7

# Introduction

Bernice’s Threaded Merge Sort (BTMS) will sort large datasets with Java using the merge sort algorithm.

### Sorting Decisions

There are several sorting algorithms to choice from. With an increase in the amount of data being created, implementing the best sort for the data will affect performance. George Heineman says in *Algorithms in a Nutshell* that “Because today’s computers are so much more powerful than the ones of 50 years ago, the size of the data sets being processed is now on the order of terabytes of information. Although you may not be called on to sort such huge data sets, you will likely need to sort large numbers of items.” Merge Sort is considered as a stable sort. It is also considered as well-suited for sorting data in secondary storage. (Heineman).

### Java Threads

To further improve the merge sort performance, multi-threaded sorts are another option. In *An Implementation of Sorting Algorithm Based on Java Multithread Technology*, Wang says

“With the spread of muticore computer, ordinary desktop computers have strongparallel processing ability. But using traditional serial sort algorithm cannot take full advantage of powerful parallel computing power of the computer. Therefore, how to reduce the recursion level and how to improve the traditional algorithms, made him able to adapt to the development of computer parallel technology, to increase the efficiency of traditional algorithms to a new level, is a worthy subject of study.”

## BTMS Proposition

BTMS will consist test the performance of a non-threaded merge sort and a multi-threaded merge sort on Windows 10 and Linux.

# Algorithms / Project Solution

This section describes the unique problem and the proposed solution that will be known as BTMS.

**The Merge-Sort Algorithm**

Merge-Sort is an algorithm that divides its input list into two lists, sorts them, and merges the two sub-lists into a sorted version of the input list.

MERGESORT(A)

ifLENGTH(A) =< 1

then returnA

B <- MERGESORT(first half of A)

C <- MERGESORT(second half of A)

A <- MERGE(B, C)

returnA

**Threads**

The multi-threaded program will allow the following number of threads to be used for the merge-sort:

* 1
* 2
* 4
* 8
* 500
* 1000

**Data Set Size**

Different data set sizes will be created and tested. An array of integers will be created using the dataset. This size may hardware dependent and will be tested up to 10 billion.

**Timing**

The time will be taken before and after the sort is performed using: System.currentTimeMillis();

# Implementation

This section describes the resources that will be used to implement BTMS.

## Platform

BTMS will be a Java 8 application. It will be installed on an Intel platform running:

* Windows 10 Operating System (OS)
* Ubuntu Linux 14.04 Bootable Flash

Information and downloads for Java can be found at <https://docs.oracle.com/javase/8/>.

## Development Tools

Development will be done on an Intel machine running Windows 10 and a Linux OS Bootable drive. I will be using NetBeans IDE for Java EE Developers. Information and downloads can be found at <http://www.oracle.com/technetwork/articles/javase/jdk-netbeans-jsp-142931.html>.

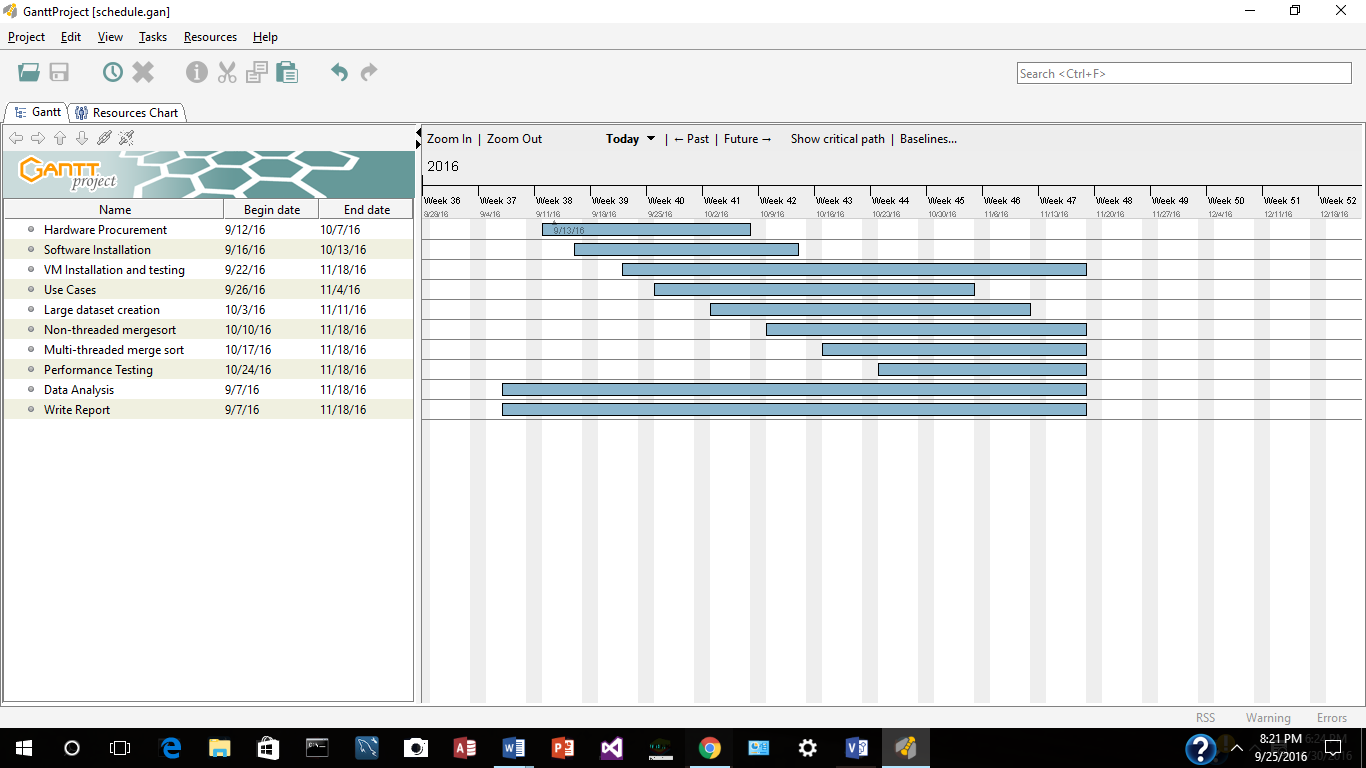
## Programming Language and Code Libraries

BTMS will be programmed in the Java programming language using the NetBeans IDE.

## Implementation Procedure

I plan to start by procuring a Windows 10 laptop, installing OS updates and installing software. I will also install Oracle VirtualBox, download a Windows 10 VM and test it during this phase. Next I will develop high-level use case diagrams for the threaded application. Also during this phase I’ll design the interface. Next, I will install Java and the NetBeans IDE on both Windows 10 hard and an Ubuntu Linux 14.04. I will research existing datasets, code to produce a dataset or writing a Java program to create a large dataset of integers to be sorted by the merge-sort programs. I will then write a non-threaded merge-sort. After the non-threaded merge-sort is complete I will write the multi-threaded merge-sort. Then I will run performance tests to compare the non-threaded merge-sort with the threaded merge-sort with different number of threads.

# Schedule



# References

Heineman, George T., et al. *Algorithms in a Nutshell*, O’Reilly Media, Sebastopol, CA, 2016.

Wang, Deming, et al*. An Implementation of Sorting Algorithm Based on Java Multithread*

*Technology*, vol. 1, IEEE, 2012.doi:10.1109/ICCSEE.2012.152.

Concurrency Tutorial by Oracle,

<http://docs.oracle.com/javase/tutorial/essential/concurrency/index.html>, Accessed in September 2016.

Java Documentation by Oracle,

<https://docs.oracle.com/javase/8/docs/api/java/lang/Thread.html>, Accessed in September 2016.

Java Documentation by Oracle,

<https://docs.oracle.com/javase/8/docs/api/java/util/Arrays.html>, Accessed in September 2016.