M.S. in Computer Engineering

CMPE180C: Operating Systems Design

Professor. Hungwen Li

PROJECT REPORT

ON

MULTITHREADED SORTING APPLICATION

TEAM 12

NEERAJ VENUGOPAL (STUDENT ID - 011541907)
HARIKA REDDY BILLURI (STUDENT ID - 012462164)
YASHASWI DODDAVEERAPPA (STUDENT ID - 012468066)

ABSTRACT:

The main aim of the project is to sort N random integers using the concept of multithreading. A list of unsorted N integers is taken as input by the system. The project uses merge sort algorithm. According to merge sort algorithm, the given set of integers is divided into two subsets and are sorted individually and then merged to generate a final sorted list of integers.

Table of Contents

APPROACH:

Sorting a list of integers using thread concept is more efficient when compared to classical approach. The set of integers are passed phase by phase and stored in an array. This array is shared amongst all other threads. A global array is passed to a merge sort algorithm which divides the array into subsets, each handled by one thread. The algorithm runs the two sorting threads in parallel and finally merges them into one single sorted array with an overall efficiency of O (n log n). The sorted array is then printed by the parent thread.

INTRODUCTION TO THREADS

Thread is the smallest unit of programmed instructions which can be managed independently by a scheduler. Thread is a component of a process. In Modern operating systems, a thread exists within a process i.e. a single process may contain multiple threads and they can also share resource and execute in parallel.

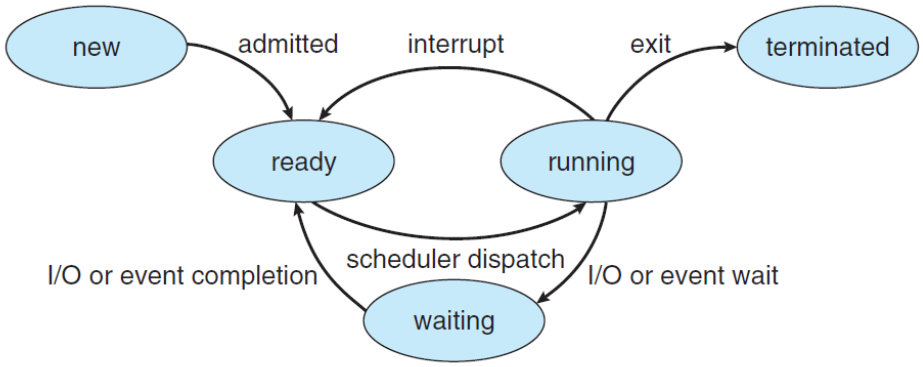
Multitasking is referred as multiple processes sharing common processing resources such as a CPU, memory etc. Multi-threading is an extended idea of multitasking where it is possible to subdivide specific operations or data within a single application into individual threads. A multi-threaded program contains two or more parts executing concurrently and each part is handled as a different task at the same time making optimal use of the available resources.

The life cycle of the thread are as follows:

NEW – The thread is newly created and has not yet started the execution also called as Born thread.

READY – The process is waiting to be assigned to a processor so that they can run.

RUNNING – Once the process has been assigned to a processor, the process state is set to running and the processor executes its instructions



WAITING – Process moves into the waiting state if it needs to wait for a resource, such as waiting for user input, or waiting for a file to become available.

TERMINATED – Once the process finishes its execution, it is moved to the terminated state where it waits to be removed from main memory.

Thread Methods used in implementation:

Amongst the various available methods in Thread class, the following three methods are used in the implementation:

run (): This method provides an entry point for the thread and complete business logic is implemented inside this method.

Syntax: public void run ().

start (): After the thread instantiation the thread method is started by calling start () method which as well executes a call to run () method.

Syntax: public void start ().

join (): The current thread invokes this method on a second thread, causing the current thread to block until the second thread terminates or the specified number of milliseconds passes.

Syntax: public final void join (long millisecond).

Advantages of Multithreading

- They have faster and quick response.
- Decreased cost of maintenance.
- Better use of CPU resource.
- Improved Performance.
- Threads within program share data with each other and hence extra space is not required.
- Low Resource Consumption.

SORTING

Sorting is ordering a list of objects i.e. to order data in an increasing or decreasing fashion according to some linear relationship among the data items. There are many techniques used for sorting and the Implementation of particular sorting technique depends upon situation. Searching of data elements can be advanced to a better high level using sorting and also data can be represented in more readable formats.

Merge Sort

Merge Sort is a popular, efficient sorting technique based on Divide and Conquer algorithm. It is based on the idea of breaking down a list into several sub-lists until each sub list consists of a single element and merging those sub lists in a manner that results into a sorted list. While comparing two sub lists for merging, the first element of both lists is taken into consideration. While sorting in ascending order, the element that is of a lesser value becomes a new element of the sorted list. This procedure is repeated until both the smaller sub lists are empty and the new combined sub list comprises all the elements of both the sub lists.

Steps to sort the elements using Merge Sort:

- **Divide Step**: If a given array has zero or one element, then it is already sorted. Otherwise, split into two subarrays, each containing about half of the elements of the array.
- **Conquer Step:** Conquer by recursively sorting the two sub arrays.
- **Combine Step:** Combine the elements by merging the two sorted subarrays into a sorted sequence.

Time Complexity

The time complexity for merging two sorted array of length n, into one sorted array of length 2n is **O (2 n).** In merge sort since we will divide array into two equal parts, later to merge it in to sorted order we will have to perform merging of array in all the levels with time complexity of **O(n)**. Therefore, for n elements we will have log (n) levels, and for each level we will perform n comparison hence time complexity is **O(nlogn).**

IMPLEMENTATION

1) Sorting Array with 10 Elements:

```
Harikas-Air:Desktop harikareddy$ javac MultiThreadSorting.java
Harikas-Air:Desktop harikareddy$ java MultiThreadSorting

Enter the Size of Array to be Sorted :10

Array to be Sorted :            |29 |50 |32 |104 |14 |26 |72 |28 |7 |0|

Two Sorted Arrays Before Merging :      |14 |29 |32 |50 |104|       |0 |7 |26 |28 |72|

Sorted Array:           |0 |7 |14 |26 |28 |29 |32 |50 |72 |104|

Time taken for MultiThreadSorting using two threads : 0.003 seconds

Exit? (Enter Y/N) : N
```

2) Sorting Array with 35 Elements:

```
Enter the Size of Array to be Sorted :35

Array to be Sorted :            |98 |19 |89 |12 |62 |68 |18 |105 |59 |92 |58 |88 |36 |59 |101 |86 |90 |20 |75 |54 |
47 |78 |81 |15 |12 |69 |82 |106 |10 |57 |75 |38 |11 |100 |0|

Two Sorted Arrays Before Merging :      |12 |18 |19 |36 |58 |59 |59 |62 |68 |86 |88 |89 |90 |92 |98 |101 |105|
 |0 |10 |11 |12 |15 |20 |38 |47 |54 |57 |69 |75 |75 |78 |81 |82 |100 |106|

Sorted Array:           |0 |10 |11 |12 |12 |15 |18 |19 |20 |36 |38 |47 |54 |57 |58 |59 |59 |62 |68 |69 |75 |75 |78
|81 |82 |86 |88 |89 |90 |92 |98 |100 |101 |105 |106|

Time taken for MultiThreadSorting using two threads : 0.001 seconds

Exit? (Enter Y/N) : N
```

3) Sorting Array with 75 Elements:

```
Enter the Size of Array to be Sorted :75


Array to be Sorted :            |78 |32 |26 |28 |65 |76 |69 |76 |72 |9 |18 |15 |52 |59 |51 |69 |44 |29 |72 |68 |19
|54 |95 |58 |69 |11 |88 |30 |72 |41 |16 |104 |21 |100 |17 |65 |76 |43 |20 |63 |33 |93 |76 |31 |81 |33 |98 |36 |49 |
34 |97 |19 |13 |93 |47 |84 |38 |11 |74 |93 |26 |80 |31 |44 |101 |62 |69 |15 |30 |52 |38 |31 |82 |82 |0|


Two Sorted Arrays Before Merging :      |9 |11 |15 |16 |17 |18 |19 |21 |26 |28 |29 |30 |32 |41 |44 |51 |52 |54 |58
|59 |65 |65 |68 |69 |69 |69 |72 |72 |72 |76 |76 |76 |78 |88 |95 |100 |104|       |0 |11 |13 |15 |19 |20 |26 |30 |31
 |31 |31 |33 |33 |34 |36 |38 |38 |43 |44 |47 |49 |52 |62 |63 |69 |74 |76 |80 |81 |82 |82 |84 |93 |93 |93 |97 |98 |1
01|


Sorted Array:           |0 |9 |11 |11 |13 |15 |15 |16 |17 |18 |19 |19 |20 |21 |26 |26 |28 |29 |30 |30 |31 |31 |31 |
32 |33 |33 |34 |36 |38 |38 |41 |43 |44 |44 |47 |49 |51 |52 |52 |54 |58 |59 |62 |63 |65 |65 |68 |69 |69 |69 |69 |72
|72 |72 |74 |76 |76 |76 |76 |78 |80 |81 |82 |82 |84 |88 |93 |93 |93 |95 |97 |98 |100 |101 |104|


Time taken for MultiThreadSorting using two threads : 0.003 seconds


Exit? (Enter Y/N) :
```

CONCLUSION:

The list of N random integers is taken as input into the system and is sorted in ascending order using merge sort algorithm. This algorithm is implemented using the concept of multithreading which enhances its performance. The output shows the list of sorted integers along with the time taken for sorting.

REFERENCES:

Operating System Concepts, Abraham Silberchatz, Peter Baer Galvin and Greg Gagne