ValarianCouch_CS627_IP3.docx

Sources Overview

31%

OVERALL SIMILARITY

1	colorado-technical-university on 2020-06-11 SUBMITTED WORKS	15%
2	colorado-technical-university on 2020-03-11 SUBMITTED WORKS	6%
3	colorado-technical-university on 2019-09-05 SUBMITTED WORKS	4%
4	colorado-technical-university on 2020-09-16 SUBMITTED WORKS	3%
5	colorado-technical-university on 2020-09-21 SUBMITTED WORKS	3%

Excluded search repositories:

- None

Excluded from Similarity Report:

- Bibliography
- Quotes
- Small Matches (less than 8 words).

Excluded sources:

- None

Part 1:
Considering the following as the set of given inputs:.

n being the files number

s[n] is the array which has the independent file size.

m is disks

t[m] is the storage on each disk

The following is the algorithm/pseudocode used for the transfer if n files to the m disks while storing the results on storage on map[i] which is the disk index.

```
Function GreedyAlgoFit (n,m,s[],t[])

        1. Declare map array of size n.
        2. Declare file array file_array[] of size n //Stores indices of
           original files
        3. Create disk array disk_array[] of size m //stores indeces of
           original disks
        4. Sort the files in descending order
        5. Sort the disks in descending order
        6. For k=0 to n and l=0 to m, execute:
              I.   If t[k]>=s[l]
             II.   Map[file_array[k]=disk_array[l]
            III.   t[l]=t[l]-s[k]
             IV.   Increment the variables
    7. Return map.
```

Part 2:
My solution provided above is implemented based on greedy algorithm. It therefore can't guarantee an optimal solution for both cases.

Estimating the running time:

- 1 and 2 steps takes O(n+m) which is O(nlogn) same as merge sort in merging the arrays
- 3 to 4 takes O(m+n) which is same s as O(n) time
- The other steps takes a constant time.
- T(n)= O(n*ogn) + O(m*logm) + O(m+n).

Part 3:
The algorithm will result to O(m*n) running time estimation in terms of big Oh. This is because brute force compares the size of every file in every space of all the disks.