Analysis of Algorithms II

BLG 336E

Homework 3

Seyyid Osman Sevgili
504221565

2023 - Spring

Contents

1 Explanation and Pseudo-code

The main function works with arguments. The argument is expected to be a case number, which is used to build a base input path for the reading of schedules, assets, and availabilities.

The first major block of the code involves reading in data for schedules, assets, and availabilities from the specified paths and storing them in appropriate data structures.

Next, is the `findBestScheduleForEachPlace` function. This function iterates over all the schedules and, for each place, it keeps the schedule with the highest revenue. The best schedules are then written to a file.

The best schedules are then converted into a series of events. An event here refers to a schedule at a particular place, with a specific start and end time. The `findBestTour` function is then used to find an optimal set of events that maximize the total revenue while ensuring that there are no overlaps between the events. The best tour is then written to a file.

Finally, the code performs a variation of the knapsack problem.

1.1 Main Funtion Pseudocode

```
Function main:
  Check number of command-line arguments
  Extract case number and formulate base input path

  Read schedules, assets, and availabilities from files

  Find and write optimal schedules

  Initialize totalRevenue and totalValue to 0
  Convert optimal schedules to events
  Find and write the best tour

  Perform knapsack operation and write the result

  Return 0
```

The time complexity of the main function is determined by the function with the highest time complexity among the called functions, which in this case is $O(n \log n)$, where n is the number of Events

1.2 Pseudocode and Time Complexity for findBestScheduleForEachPlace

```
Function findBestScheduleForEachPlace(schedules):   Time Complexity
  Initialize an empty map of schedules  O(1)

  For each schedule in the schedules list:  O(n) for the loop
    If the place of the schedule is not in the map:  O(1)
      Add the schedule to the map   O(1)
    Else:
      Compare the current schedule with the existing one in the map   O(1)
      If the current schedule is better:
        Replace the schedule in the map   O(1)

  Return the map  O(1)
```

The overall time complexity is O(n), where n is the number of schedules.

1.3 Pseudocode and Time Complexity for findBestTour

```
Function findBestTour(events, totalRevenue):    Time Complexity
  Sort events in non-decreasing order of their end times    O(n log n)

  Initialize an empty list for the best tour    O(1)
  Initialize a variable for the current time    O(1)

  For each event in the events list:    O(n) for the loop
    If the event start time is greater than the current time:   O(1)
      Add the event to the best tour    O(1)
      Update the current time to the end time of the event  O(1)
      Add the revenue of the event to totalRevenue  O(1)

  Return the best tour  O(1)
```

The overall time complexity is O(n log n), where n is the number of events.

1.4 Pseudocode and Time Complexity for knapsack

```
Function knapsack(assets, totalRevenue, totalValue):    Time Complexity
  Sort assets in non-increasing order of their value/weight ratio   O(n log n)
```

```
Initialize an empty list for the asset list   O(1)

For each asset in the assets list:    O(n) for the loop
  If the asset can be fully added to the asset list   O(1)
  without exceeding the totalRevenue:
    Add the asset to the asset list   O(1)
    Subtract the weight of the asset from totalRevenue    O(1)
    Add the value of the asset to totalValue  O(1)
  Else if a fraction of the asset can be added to the asset list: O(1)
    Add the fraction of the asset to the asset list   O(1)
    Update totalRevenue and totalValue accordingly    O(1)

Return the asset list O(1)
```

The overall time complexity is O(n log n), where n is the number of assets.

2 Questions

2. In the homework, you are given two restrictions;
(a) You need to be in one place for the whole day (For example you can't be in place A for 9:00-13:00 and place B for 13:30-17:00 in one day.)
(b) You can not break the determined available time intervals (For example you can't be in place A for 15.04-17.04 and be in another place after 17.04 if place A is available for the time period 15.04-19.04. You have to be in place A for the full period.)

2.1 How would the result be if you discard these constraints for the case 1? First, break the second restriction. Then, discard the first one also.

(b) If we ignore the rule that says we can't leave a place before its available time ends, we might be able to make more money. For instance, Imagine two places are available at the same time and you can earn money at both. If the total money you can make from both places during the overlapping time is more than what you can make from staying at one place the whole time, it's better to spend some time at each place.
(a) If we also ignore the rule that says we have to stay in one place for the whole day, we could potentially make even more money. This is because we could visit many places in one day, increasing our earnings. For example, imagine we have several places to visit and each place gives us money. If we don't have to stay in one place all day, we can move around to different

places. By doing this, we can add up all the money we make from each place, which could be more than what we could make from staying at one place the whole day.