Comp 3121 Algorithms & Programming Tech - Question 3

z5335039 Chen En

Question 3 *CSE Labs*

[30 marks] The School of CSE is scheduling some labs for its classes. There are n classes that need to be scheduled into one of the n available labs, and they would like all of the labs to run at the same time. A single class cannot occupy multiple labs, and multiple classes also cannot share a lab.

3.1 [4 marks] For a class to use a lab, there has to be at least 1 seat for each student. Given an array $C[1...n]$ where $C[i]$ indicates the number of students in class i, and an array $L[1...n]$ where $L[j]$ indicates the number of seats in lab j, write an $O(n \log n)$ time algorithm that assigns a lab room to each class. Your algorithm should also identify if an assignment is not possible.

3.2 [7 marks] It turns out that classes are very picky about which labs they want. In particular, you are now provided an additional array $B[1..n][1..n]$ with

$$B[i][j] = \begin{cases} \texttt{True} & \text{class } i \text{ likes lab } j \\ \texttt{False} & \text{otherwise} \end{cases}$$

Given $C[1..n]$, $L[1..n]$ and $B[1..n][1..n]$, write an $O(n^3)$ algorithm to assign a lab room to each class. Your algorithm should also identify if an assignment is not possible.

3.3 [12 marks] Unfortunately an issue with the building has affected the power and now there are only $m < n$ labs available. The school has no choice but to schedule some classes at different times. Given the arrays $C[1...n]$, $L[1...m]$, $B[1..n][1..m]$ defined as in the previous parts, write an $O(n^3)$ time algorithm that determines whether it is possible to schedule the classes with T different time slots available.

3.4 [7 marks] Each of the classes also requires one of k tutors to run the lab. Each tutor is only allowed to run a maximum of 3 labs, and can only run 1 lab at a time. The tutors' availability is given as an array $D[1..k][1..T]$ with

$$D[i,j] = \begin{cases} 1 & \text{if the } i\text{th tutor is available at timeslot } j \\ 0 & \text{otherwise} \end{cases}$$

For some number of timeslots $T < n$, write an $O(n^3)$ algorithm that assigns a room, time, and tutor to each lab if a schedule is possible.

3.1
Apply merge sort to both array C and L then store the after-sorting elements into array C' and L' separately. Then we compare each same index element from array C' and L', starting from the last index n, n-1,…,1, 0. Given by the question, one class can only be paired with one lab, and the numbers of seats must be greater than the number of students. Since we compare from the last index elements from two sorting arrays(C'[n], L'[n]). If $C'[n] > L'[n]$, there is no element in array L' can be greater than C' because the rest of the elements in L' are all smaller than $L'[n]$. This method applies the greedy algorithm which ensures we always get the optimal pair as every single matching.

To determine whether the assignment is possible, we check every index i which $0 \leq i \leq n$.
Case1: If $C'[i] > L'[i]$, which means the assignment is not possible since there are not enough seats for the students.
Case 2: If $C'[i] \leq L'[i]$, it means there are more seats than the students' number so the assignment is possible.

Time complexity
The time complexity of sorting two arrays takes O(2nlogn), and the comparing of elements from two arrays takes O(n). Thus, the time complexity for this operation takes $O(2nlogn) + O(n) = O(nlogn)$.

3.2

Construct a flow network with:

1. Source S and sink T.
2. A layer of n vertices for classes, for each class $C_i \; that \; 1 \leq i \leq n$.
3. A layer of n vertices for labs, for each lab $L_j \; that \; 1 \leq j \leq n$.
4. If $B[i][j]$ = true and $C[i] < L[j]$, we construct an edge from C_i to L_j with capacity 1. The statement presents that class i likes lab j and there is enough capacity of seats for the students, and the capacity of 1 ensures one class can only choose one lab. There are at most n^2 edges if each class C_i can be matched with all n labs.
5. From source to each class C_i, there is an edge with capacity 1 since each class can only choose one lab. There are total n edges.
6. From each lab L_j to the sink, there is an edge with capacity 1 since each lab can only be chosen by one class. There are total n edges.

Consider the flow in this graph:

1. The paths from source S to T present the choosing of labs from classes. If there is a flow from C_i to L_j which means the class i choose the lab j.
2. The maximum flow F $(F \leq n)$ presents the number of classes that have successfully chosen the labs

To determine whether the assignment is possible, we find the maximum flow by applying Ford-Fulkerson's Algorithm to the flow network.

case1: If the maximum flow $F = n$, which means all n classes have successfully matched with n labs, so the assignment is possible.

case2: If the maximum flow $F < n$, which means only F classes have chosen the labs.

Time complexity

There are at most $n^2 + 2n$ edges, and the max flow is bounded by n. By applying Ford-Fulkerson's Algorithm, the time complexity is $O(E|f|) = O(n * (n^2 + 2n)) = O(n^3)$.

3.3
Construct a flow network with:

1. Source S and sink T.
2. n vertices for classes, for each class C_i $that$ $1 \leq i \leq n$.
1. m vertices for labs, for each lab L_j $that$ $1 \leq j \leq m$.
2. If $B[i][j] = true$ and $C[i] < L[j]$, we construct an edge from C_i to L_j with capacity 1. The statement presents that class i likes lab j and there is enough capacity of seats for the students, and the capacity of 1 ensures one class can only choose one lab. There are at most nm edges if each class C_i can be matched with all m labs.
3. From source to each class C_i, there is an edge with capacity 1 since each class can only choose one lab. There are total n edges.
4. From each lab L_j to the sink, there is an edge with capacity T since each lab has T timeslot to be chosen. There are total m edges.

Consider the flow in this graph:

1. The paths from source S to T present the choosing of labs from classes. If there is a flow from C_i to $L_{j(in)}$, in which means the class i choose the lab j. The timeslot t_l receives the flow from L_j when lab j can be run the timeslot l.
1. The maximum flow F $(F \leq n)$ presents the number of classes that have successfully chosen the labs

To determine whether the assignment is possible, we find the maximum flow by applying Ford-Fulkerson's Algorithm to the flow network.
case1: If the maximum flow $F = n$, which means all n classes have successfully matched with m labs, so the assignment is possible.
case2: If the maximum flow $F < n$, which means only F classes have chosen the labs with suitable timeslot.

Time complexity
There are at most $nm + m + n < n^2 + 2n$ edges because $m < n$, and the max flow is bounded by n. By applying Ford-Fulkerson's Algorithm, the time complexity is $O(E|f|) = O(n * (nm + m + n)) = O(n^3)$.

3.4
Construct a flow network with:

1. Source S and Sink T.
2. n vertices for classes, for each class C_i $that$ $1 \leq i \leq n$.
3. m vertices for labs, for each lab L_j $that$ $1 \leq j \leq m$.
4. T vertices for time slots, for each time slot t_l that $1 \leq l \leq T$.
5. k vertices for tutors, for tutor T_r $1 \leq r \leq k$.
6. If $B[i][j]$ = true and $C[i] < L[j]$, we construct an edge from C_i to L_j, in with capacity 1. The statement presents that class i likes lab j and there is enough capacity of seats for the students, and the capacity of 1 ensures one class can only choose one lab. There are at most $n * m$ edges if each class C_i can be matched with all m labs.
7. We construct an edge of capacity 1 between Li to each time slot t_l since every lab can be run in T different time slots. There are $m * T$ edges.
8. If $D[r][l]$ = true which means the tutor r is available at that time slot, we construct an edge of capacity 1 between tl and T_r. There are at most $k * T$ edges.
9. From source S to each class C_i, there is an edge with capacity 1 since each class can only choose one lab. There are total n edges.
10. From each tutor T_r to the sink T, there is an edge with capacity 3 since each tutor is only allowed to teach at most three labs. There are total k edges.

Consider the flow in this graph:

1. The paths from source S to T present the assignment of classes to labs, timeslot then to which tutor. If there is a flow from C_i to L_j, in which means the class i choose the lab j. The timeslot t_l receives the flow from L_j when lab j can be run the timeslot l. The tutor T_r receives the flow from t_l when tutor r is available to teach the lab at time slot l.

2. The maximum flow F ($F \leq n$) presents the number of classes that have successfully chosen the labs with suitable time slot and tutor to teach.

To determine whether the assignment is possible, we find the maximum flow by applying Ford-Fulkerson's Algorithm to the flow network.
case1: If the maximum flow $F = n$, which means all n classes have successfully matched with m labs, so the assignment is possible.
case2: If the maximum flow $F < n$, which means only F classes have chosen the labs.

Time complexity
The max flow is bounded by n, and there are at most $(k + n + mT + mn + 3k) < n^2$ edges because n is greater than m, T, and k. By applying Ford-Fulkerson's Algorithm, the time complexity is $O(E|f|) = O(n * (k + n + mT + mn + 3k)) = O(n^3)$.