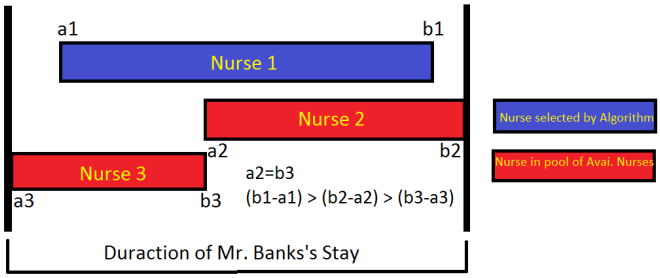
Stephen Woodbury 4/30/17 CMPS 102
I have read and agree to the collaboration policy. Stephen Woodbury.
Collaborators: none

Assignment 2_3 : Greedy Algorithm

a) Disproving one Greedy Strategy:



If the algorithm described in "a" is used, we have the potential to build a set of nurses that isn't the smallest it could be. An example is given above.

b) Creating a Greedy Algorithm:

Components: N : Array of Structs(struct = [#, a, b]) of size n.
TP: Total Number of nurses we can draw on (TP <= n).
LI: Last Nurse (Index) worked on
possible: bool, true if solution is possible, false otherwise.
covered: bool, true if solution found, false otherwise.
found: bool, true if nurse found for startTime, false otherwise
startTime: int, start Time of time left to cover
finishTime: int, finish Time of Bank's stay at hospital
S: Queue of Nurses selected for Solution

Algorithm:

1) *initialize: possible=true, covered=false, found=false, finishTime=End time of period Banks needs covering for, startTime=0, LI=0, S=nothing, TP=0*
2) *Sort List of Pairs into N in order of earliest "a" to latest "a". If a tie occurs, take the pair with the max "b". N.#=pair #, N.a=a, N.b=b. Increment TP by 1 every time an array slot is filled in. Will need to use LI and increment accordingly, set back to 0 when all is sorted.*

```
3) while(possible && !covered)
4)   found = false;
5)   for(int i = LI; i>=0 && !found; i--)
6)     if( N[i].a > startTime )
7)       possible = false;
8)     else
9)       if(N[i].b > startTime)
10)        S.push(N[i].#);
11)        possible = true;
12)        startTime = N[i].b;
13)        if(startTime >= finishTime)
14)          covered = true;
15)        found = true;
16)        while(N[i].a < startTime && i<TP)
17)          i++;
18)        LI = i;
19) if(possible)
20)   return S;
21) else
```

22) count<<"no such subset of nurses exist";
23) return S;

Algorithm Description:

Algorithm starts by initializing all relative values and sorts our pair of lists into our array of structs. Then the algorithm chooses the next nurse to work on(the first nurse in the array if at the start) and checks the start time of said nurse to needed start time. If nurse start time is greater than needed start time, look at nurses that start at an earlier time that haven't been allocated yet, check if their end time is later than the start time needed. If no nurse is found that matches these requirements, no solution exists. If one of these requirements exist, take said nurse and add it to solution queue. Take end time of said nurse, set it to new startTime, change LI (next nurse to work on) to be the nurse that has a start time that is greater than or equal to said startTime. Check to see if the duration has been filled, if it has, solution has been found, return S.

c) Algorithm Correctness:

Proof of Correction : Termination:

Claim: Our while loop terminates after at most n calls.

Pf: Every time the while loop is called, one new nurse is added to our solution queue or the solution is not possible and the while loop won't run again. If all nurses have been looked at, either all nurses have been added to the solution queue or there is no solution and the while loop won't run again. There are only n-nurses to add.

Proof of Correction : Optimality:

Claim: Our Greedy algorithm G is the most optimal.

Pf (By contradiction): Assume C is our most optimal algorithm. The solution provided by G is: NG1, NG2, ..., NGr,..., NGk Where NG# stands for Nurse Greed choice #. The solution to C is: NC1, NC2, ..., NCr,..., NCk' where NC# stands for Nurse Optimal choice #. NG1=NC1, NG2=NC2,..., NGr=NCr. Greedy and Optimal algorithm have at most their first r nurse choices similar. The r+1 choice of Optimal differs from Greedy's r+1 choice. Optimal's r+1 choice either 1) must have an earlier or equal start time to Greedy's r+1 choice. But since our r+1 choice for Greedy is a valid choice, we can sub optimal's choice with Greedy's choice with no cost or 2)the end time of Optimal's choice is shorter or equal to Greedy's choice where since Greedy's choice is just as valid, Optimal's choice can be swapped with Greedy's choice at no cost. Either way, there is a contradiction as now Optimal and Greedy now share r+1 Nurse choices. QED

d):Running Time Complexity:

Time Complexity:

1) O(1)
2) O(nlogn)
3) O(n)
4) O(1)
5) O(n)
6-15) O(1)
16)O(n)
17-23)O(1)

Total: O(1)+O(nlogn)+O(n)(2*O(n)+13*O(1))+2*O(1) =* ***O(n^2)***

Space Complexity:

N : O(n), TP,LI,possible,covered,found,startTime,finishTime : O(1), S:O(n)

*total: 2*O(n)+7*O(1) =* ***O(n).***