COL 333

Assignment 2

Satwik Jain, 2019CS10398

Shubh Sahu, 2019CS10403

We were supposed to create a nurse rostering system.

We begin by taking input of a test case, first we check for two conditions :

1. M+A+E>6\*N/7 : N-M-A-E gives the value of R. the following constraint should satisfy so that each nurse can get atleast one rest in a week 7\*R>N. combining both, we get the above mentioned condition.

2. 2\*M+E >N : the number of nurses getting morning or evening on a day is M+E. these nurses are unavailable to take on morning shift for next day so there should be atleast N-M shift present for these M+E nurses giving above condition.

If any of the above two inequality is satisfied, we can conclude that no solution is possible in current case.

Now We will talk about implementation of the problem:

In the **first part**, we had to follow the hard constraints. We initiated an array that stored the assignments for each nurse on a particular day. We began by initiating the nurses for the first day.

Then we started the recursive backtracking from the second day.

We created the following helping functions:

1.**NurseSolver()** – This is the main recursive backtracking function. In this function, first we check whether the current assignment till now is valid or not by validfurther function, If the assignment is valid, it selects the best value of assignment for that nurse from the valueselection function.

If assignment of the particular value to our current assignment is valid according to the constraints, it assigns the value and recursively calls the nurseSolver function for the next nurse on the same day or the first nurse on next day if current nurse was the last nurse.

If the value obtained from the value selection function is not a valid assignment according to checkvalid, that value becomes unavailable and then we find the next most suitable value using valueselection function.

2. **validfurther()** – we know that morning cannot be assigned to a nurse if its assignment on previous day was morning or evening. Validfurther function finds the number of nurses at any current point which were assigned morning or evening yesterday and correspondingly finds the number of nurses which can be assigned morning today. If that value is less than the current value of morning slots remaining this means there are more slots than the nurses which can be assigned so it returns false.

3. **Checkvalid**() – checkvalid is a function which takes care of the constraints which are to be applied. It checks for the following things:

i. current assignment is morning and previous assignment was morning or evening.

ii. we have an assignment but there are no slots available for that particular assignment

iii. For any particular day, if we are at the last nurse and have an assignment there should be no slot possible for any other type of assignment

4. **Valueselection** – Valueselection function is the which selects a possible assignment according to the predefined preferences. First we check if we have encountered a rest for that nurse in the current week, if we have not encountered a rest, we prefer giving a rest to the nurse. If a rest has been encountered we then give preference to evening followed by afternoon and then morning and rest.

For the **Second Part**, we simply made some minor adjustment to the way we find the value for the senior nurses. In this part, for the senior nurses, we first check if it is the last day of the week. If it is the last day of the week and we have not given a rest yet we give them rest otherwise we give preference to morning and evening followed by other two. While for the normal nurses, we keep the same preference model as for the previous part.