VERSIÓ SENSE TOLERÀNCIA

4. You've periodically helped the medical consulting firm Doctors Without Weekends on various hospital scheduling issues, and they've just come to you with a new problem. For each of the next n days, the hospital has determined the number of doctors they want on hand; thus, on day i, they have a requirement that *exactly* p_i doctors be present.

There are k doctors, and each is asked to provide a list of days on which he or she is willing to work. Thus doctor j provides a set L_j of days on which he or she is willing to work.

The system produced by the consulting firm should take these lists and try to return to each doctor j a list L'_j with the following properties.

(A) L'_j is a subset of L_j, so that doctor j only works on days he or she finds acceptable.

(B) If we consider the whole set of lists $L'_1, \ldots, L'_k$ it causes exacty p_i doctors to be present on day i, for $i = 1, 2, \ldots, n$.

(a) Describe a polynomial-time algorithm that implements this system. Specifically, give a polynomial-time algorithm that takes the numbers $p_1, p_2, \ldots, p_n$, and the lists $L_1, \dots, L_k$, and does one of the following two things.

- Return lists $L'_1, L'_2, \dots, L'_k$ satisfying properties (A) and (B); or
- Report (correctly) that there is no set of lists $L'_1, L'_2, \dots, L'_k$ that satisfies both properties (A) and (B).

Versió amb tolerància c

(b) The hospital finds that the doctors tend to submit lists that are much too restrictive, and so it often happens that the system reports (correctly, but unfortunately) that no accecptable set of lists $L'_1, L'_2, \dots, L'_k$ exists.

Thus the hopspital relaxes the requirements. They add a new parameter $c > 0$ and the system now should try to return to each doctor j a list L'_j with the following properties.

(A*) L'_j contains at most c days that do not appear on the list L_j.

(B) (*Same as before*) If we consider the whole set of lists $L'_1, \dots, L'_k$ it causes exacty p_i doctors to be present on day i, for $i = 1, 2, \dots, n$.

Describe a polynomial-time algorithm that implements this revised system. It should take numbers $p_1, p_2, \dots, p_n$, the lists $L_1, L_2, \dots, L_k$, and the parameter $c > 0$, and do one of the following things.

- Return lists $L'_1, L'_2, \dots, L'_k$ satisfying properties (A*) and (B); or
- Report (correctly) that there is no set of lists $L'_1, L'_2, \dots, L'_k$ that satisfies both properties (A*) and (B).

Solution:

1. Since we have no limit on how many days a doctor is willing to work, there is a simple solution to this problem: For each day i, let q_i be the number of doctors that are willing to work on this day. If $q_i < p_i$, clearly there is no possible assignment; otherwise, just arbitrarily pick p_i from those q_i doctors to work on day i.

Lesson here: not everything is network flow. Always observe the special structure and special properties of the problem before solving it.

2. We do the same thing as in part (1). The difference is for day i suppose $q_i < p_i$, then we have a deficiency. Let $d_i = \max(0, p_i - q_i)$. Construct a network where we have one node u_i for each doctor i, and node v_j for each day j. For each doctor i add an edge (s, u_i) with capacity c, and for each day j add an edge (v_j, t) with capacity d_j. For each pair (u_i, v_j), add an edge of capacity 1 if doctor i do *not* want to work on day j. There is a feasible assignment if and only if the maximum flow in this network saturates all incoming edges of t.

The proof is simple. For each day j we need d_j extra doctors, and these doctors can only be the doctors that do not want to work on day j, and each doctor can only work on no more than c such days. Therefore if we split each doctor into c identical copies and each day j into d_j identical copies, then the problem turns into finding a bipartite matching of the doctors and the days such that all days need to be matched. The corresponding network flow formulation is what we had in the previous paragraph.