Pablo Acuna
CSCI 4020
Anshelevich

Computer Algorithms Homework 6

19. 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 exactly pi doctors to be present on day i, for i = 1, 2, . . . , 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,..., p_n, and the lists L_1,...,L_k, and does one of the following two things.

– Return lists $L'_1, L'_2, \ldots, L'_k$ satisfying properties (A) and (B); or

– Report (correctly) that there is no set of lists $L'_1, L'_2, \ldots, L'_k$ that satisfies both properties (A) and (B).

Solution: Since there is no lower/upper bounds to the days a doctor can work we can just loop through each day, if there are at least p_i doctors availible to work that day, we can just choose randomly p_i doctors to work that day. If we go through all the days and there are always at least the amount required, there is a valid matching and we can return our lists we've built up. If at some point we do not have enough doctors for one day, we return no valid matching.

(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 acceptable set of lists $L'_1, L'_2, \ldots, L'_k$ exists.

Thus the hospital relaxes the requirements as follows. 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 $L'_1, L'_2, \ldots, L'_k$, it causes exactly p_i doctors to be present on day i, for i = 1, 2, ..., n.

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

– Return lists $L'_1, L'_2, \ldots, L'_k$ satisfying properties (A) and (B); or

– Report (correctly) that there is no set of lists $L'_1, L'_2, \ldots, L'_k$ that satisfies both properties (A^*) and (B).

Set-Up: Let q_i be the number of doctors that are willing to work on this day. Now we shall create a directed graph G with nodes u_i for all i doctor. We will also create nodes v_j for all days n days. Now we shall have a source node, s, connect to all doctors (s, u_i), these edges going into the doctors will have a capacity of c. Now for every day a doctor doesn't request to work on day j, we shall have an edge (u_i, v_j), this edge will have capacity 1. Now we have a sink node, t, and we will connect the days to the sink, (v_j, t), with capacity max(0, p_j - q_j).

Solution: If and only if all edges going into t are saturated in the max flow then we have a feasible solution.

Proof: For each day j we need max(0, pj - q_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 copies and each day j into max(0, pj - q_j) copies, then the problem turns into finding a bipartite matching of the doctors and the days such that all days need to be matched.

Pablo Acuna
CSCI 4020
Anshelevich

Computer Algorithms Homework 6

21. You're helping to organize a class on campus that has decided to give all its students wireless laptops for the semester. Thus there is a collection of n wireless laptops; there is also have a collection of n wireless access points, to which a laptop can connect when it is in range.

The laptops are currently scattered across campus; laptop l is within range of a set S_l of access points. We will assume that each laptop is within range of at least one access point (so the sets Sl are nonempty); we will also assume that every access point p has at least one laptop within range of it.

To make sure that all the wireless connectivity software is working correctly, you need to try having laptops make contact with access points in such a way that each laptop and each access point is involved in at least one connection. Thus we will say that a test set T is a collection of ordered pairs of the form (l, p), for a laptop l and access point p, with the properties that

(i) If (l, p) $\in$ T,then l is with in range of p (i.e., p $\in S_l$).

(ii) Each laptop appears in at least one ordered pair in T.

(iii) Each access point appears in at least one ordered pair in T.

This way, by trying out all the connections specified by the pairs in T, we can be sure that each laptop and each access point have correctly functioning software.

The problem is: Given the sets S_l for each laptop (i.e., which laptops are within range of which access points), and a number k, decide whether there is a test set of size at most k.

(a) Give an example of an instance of this problem for which there is no test set of size n. (Recall that we assume each laptop is within range of at least one access point, and each access point p has at least one laptop within range of it.)

Solution: Say we have 3 laptops, laptop 1 is in range of access point 1 and 2. Laptop 2 is in range of access point 3. And laptop 3 is also in range of access point 3. This configuration of laptops and access points only allows us to have a test set of size 4. The set will look as follows...

T = {(laptop 1, access point 1), (laptop 1, access point 2), (laptop 2, access point 3), (laptop 3, access point 3)}

Removing any of the following pairs will violate one of the three conditions above therefore there is no test set of size n.

(b) Give a polynomial-time algorithm that takes the input to an instance of this problem (including the parameter k) and decides whether there is a test set of size at most k.

Set-Up: We shall create a directed graph, G. Each laptop will be a node u_i in the graph. We will also create a node for all access points v_j. We will have an edge connecting the laptops and access points, (u_i, v_j), if laptop i is in range of access point j. This edge will have a capacity of size 1 since we dont want to double count an edge in the pairing. Next we will create a source and a gadget node, s and a respectively. We will create an edge, (s, a) of size k. Now we will create an edge, (a, u_i) for all i and its capacity will be $\lceil k/n \rceil$. Now we will create our sink node, t, and have every access point have an edge, (v_j, t) with a capacity of $\lceil k/n \rceil$.

Solution: There is a feasible solution if and only if in the max flow all incoming edges to the laptops are saturated as well as all incoming edges to t are saturated and the max flow is k.

Proof: Max flow $\rightarrow$ feasible solution; we have k in range pairs, every laptop is in the set as well as every access point. Therefore, every constraint is satisfied.

Feasible soution $\rightarrow$ Max flow; we can construct the matching graph it does not violate the way we constructed the max flow reduction.