CSCI 570 - Spring 2021 - Main Exam 2

May 7, 2021

1 Long Questions

Q17 You have a company, and there are n projects and n workers. A project can only be assigned to one worker. Each worker has capacity to work on either one or two projects. Each project has a subset of worker as their possible assignment choice. We also need to make sure that there is at least one project assigned to each worker.

Give a polynomial time algorithm that determines whether a feasible assignment of project to workers is possible that meets all the requirements above. If there is a feasible assignment, describe how your solution can identify which project is assigned to which worker.

Solution:

Method 1:

You can consider this problem as a Bipartite matching problem:

Construct a flow network as follows:

- Create source s and sink t; for each project i create a vertex p_i, for each worker j create a vertex w_j;

- Connect s to all p_i with an edge, make the edge capacity equal to 1; connect all w_j to t with an edge, and the edge capacity is 1 .

- If worker j is one of project i's possible assignment choices, add an edge from p_i to w_j with edge capacity 1.

(10 points)

Algorithm:

If we can find a max-flow, $f = n$, then there is a feasible assignment. Since for each edge, the edge capacity is 1, this algorithm can be done in polynomial time. If there is one, for each p_i and w_j, check whether the flow from p_i to w_j is 1, if yes, assign project i to worker j.

(5 points)

Method 2:

You can use circulation to solve this problem.

Construct a flow network as follows:

- For each project i create a vertex p_i, for each worker j create a vertex w_j;

- If worker j is one of project i's possible assignment choices, add an edge from p_i to w_j with lower bound and upper bound 1.

- Create a super source s, connect s to all p_i with an edge of lower bound and upper bound 1.

- Create a super sink t, connect all w_j to t with an edge of lower bound 1 and upper bound 2.

- Connect t to s with an edge of lower and upper bound n.

(10 points)

Algorithm:

Find an integral circulation of the graph, because all edges capacity and lower bound are integer, this can be done in polynomial time. If there is one, for each p_i and w_j, check whether the flow from p_i to w_j is 1, if yes, assign project i to worker j.

(5 points)

Grading:

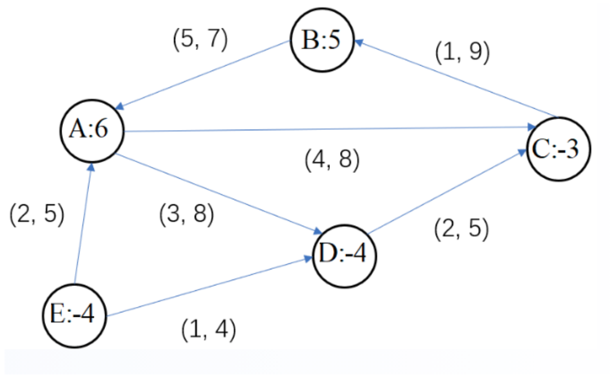
In this question, we need to make sure that there is at least one project assigned to each worker. If you remove all the redundant information, you will find it's a Bipartite matching problem.

- Construct the correct flow network: vertices, edges, and edges capacity (with lower bound and upper bound): 10 points; If the edge capacity equals to 2 (without lower bound), deduct 4 points.

- If the algorithm includes finding an integral circulation of the graph, or finding the max-flow which equals to n: 3 points.

- Describe how the solution can identify which project is assigned to which worker (check whether the flow from p_i to w_j is 1): 2 points

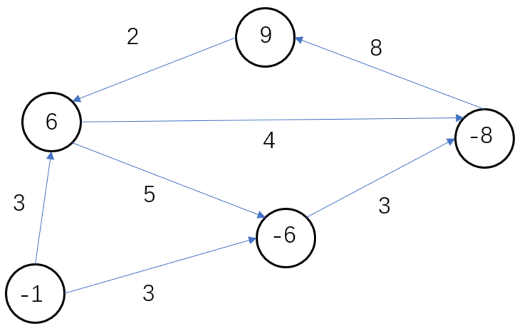
Q18 In the network below, the demand values are shown on vertices (supply values are negative). Lower bounds on flow and edge capacities are shown as (lower bound, capacity) for each edge. Determine if there is a feasible circulation in this graph. Please complete the following steps.



(1) Remove the lower bounds on each edge. Write down the new demands on each vertex A, B, C, D, E,in this order.

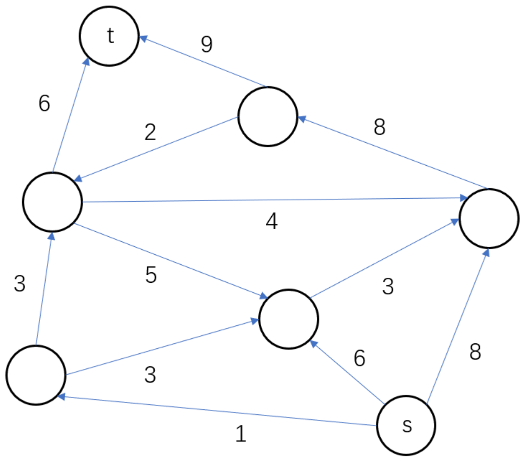
A: 6, B: 9, C: -8, D: -6, E: -1. (5 points)

(2) Solve the circulation problem without lower bounds. Write down the max-flow value.



The max-flow value is: 9. (5 points)

(3) Is there is a feasible circulation in the original graph? Explain your answer.



Check if there is a $s - t$ flow $|f| = 15$. The max-flow of this flow network is 9, so there is no feasible circulation. (5 points)

Grading:

(1) If all values are correct: 5 points, otherwise, deduct 1 point for an incorrect value.

(2) If the max-flow is correct: 5 points, otherwise, 0 points.

(3) If the answer shows that there is no feasible circulation and the explanation is reasonable: 5 points.

Q22 In a country, there are N cities, and there are some undirected roads between them. For every city there is an integer value (may be positive, negative, or zero) on it. You want to know, if there exists a cycle (the cycle cannot visit a city or a road twice), and the sum of values of the cities on the cycle is equal to 0. Note, a single vertex is not a cycle. Prove this problem is NP-Complete. Use a reduction from the Hamiltonian cycle problem. Complete the following five steps.

(1) Show that the problem belongs to NP (2 points).

The solution of the problem can be easily verified in polynomial time, just check if the sum of the values on the cycle is equal to 0, and the cycle doesn't visit a city or a road twice. Thus it is in NP.

(2) Show a polynomial time construction using a reduction from Hamiltonian cycle (6 points).

Given a Hamiltonian cycle problem. It asks if the graph exists a cycle go through all the vertices of the graph and doesn't visit a vertex twice. Let the number of vertices be N. Then we construct an instance of the zero-cycle problem (our problem), as, we set the value of one vertex to be $N-1$, and the value of any other vertices ($N-1$ vertices) to be -1.

(3) Write down the claim that the Hamiltonian cycle problem is polynomially reducible to the original problem. (2 points)

The given graph G has a Hamiltonian cycle if and only if the there is a cycle in the zero-cycle problem's graph G' and the sum of values of the cities on the cycle is equal to 0.

(4) Prove the claim in the direction from the Hamiltonian cycle problem to the reduced problem. (3 points)

If we have a solution of the Hamiltonian cycle problem, then the total value of the cycle will be $(N-1)+(-1)*(N-1)=0$, so the zero-cycle problem can also find a solution.

(5) Prove the claim in the direction from the reduced problem to the Hamiltonian cycle problem. (3 points)

If we have a solution of the zero-cycle problem, we want to prove that it's indeed a solution of the Hamiltonian cycle problem, i.e. we want to prove it will visit all vertices. If the vertex doesn't visit the vertex that has value $N-1$, then all the value on the cycle will be -1, so the total value will be negative, will not be 0, so it's impossible. So the cycle must visit the vertex that has value $N-1$. Then, in order to make the total value to be 0, it must visit all the rest of the vertices ($N-1$ vertices), since the value of them are all -1. So it will visit all the vertices. So it is a solution of the Hamiltonian cycle problem.

Grading: For (2), there may have various value assignment plans for the vertices. If the plan can make sure that, any cycle that sums up to 0 must pass all the vertices, it's correct.

Q27 The edge-coloring problem is to color the edges of a graph with the fewest number of colors in such a way any two edges that share a vertex have different colors . You are given the algorithm that colors a graph with at most $d+1$ colors if the graph has a vertex with maximum degree d. You do not need to know how the algorithm works. Prove that this algorithm is a 2-approximation to the edge coloring problem. You may assume that $d \geq 1$.

Maximum degree of the graph is d. This implies there exists at least one vertex (say v_1) connected to d other vertices. All the d edges connected to v_1 would have to be of different colors to satisfy the edge coloring. Therefore, the lower bound on the optimal solution is $OPT \geq d$.

7 points for making the above claim.

Our algorithm returns a coloring of at-most $d+1$ colors. Hence, the upper bound of the algorithm is $ALG \leq d+1$.

The approximation ratio ρ would be the upper-bound of algorithm divided by the lower-bound of the optimal algorithm which gives:

$$\rho = \frac{d+1}{d}$$

5 points for arriving at above equation.

Since, $d \geq 1$ and ρ gets largest value with $d = 1$, we have:

$$\rho = 1 + \frac{1}{d} \leq 1 + 1 = 2$$

That is, in the worst case we have a 2-approximation ration. Hence, it is a 2-approximation algorithm.

3 points for arriving at the final ratio of 2

Detailed Rubric:

(a) If only lower-bound is correct (i.e. $OPT \geq d$) but approximation ratio computation is not, award 7 points.

(b) If only upper-bound is correctly used, but lower-bound is incorrect, give 5 points. Since approximation ratio cannot be computed, that part receives 0/3.

Q21 A clique in a graph $G = (V, E)$ is a subset of nodes $C \subseteq V$ s.t. each pair of nodes in C is adjecent, i.e., $\forall u, v \in C, (u, v) \in E$. We are interested in computing the largest clique (i.e., a clique with max. no. of nodes) in a given graph. Write an integer linear program that computes this.

Let x_u be an integer variable that gets a value 1 if u is included in the clique, and 0 otherwise. The ILP formulation is as follows:

$$\begin{array}{ll} \text{max:} & \sum_{u \in V} x_u \\ \text{subject to:} & x_u + x_v \leq 1 \quad \forall\ (u, v) \notin E \\ & x_u \in \{0, 1\} \quad \forall\ u \in V \end{array}$$

Alternatively, you could associate the integer variables with the exclusion of the variables instead of inclusion, i.e., each x_u is 1 if u is excluded, and 0 if included. Then, the ILP is

$$\begin{array}{ll} \text{min:} & \sum_{u \in V} x_u \\ \text{subject to:} & x_u + x_v \geq 1 \quad \forall\ (u, v) \notin E \\ & x_u \in \{0, 1\} \quad \forall\ u \in V \end{array}$$

Grading:

- **2.5 points for correctly defining the variables and mentioning they're binary.**
- **5 points for the objective.**
- **7.5 points for the constraint.**
- **Note that having variables/constraints in addition to the ones required, often makes the solution incorrect.**
- **Partial points for any cases not confirming to the standard solution will be handled on a per-case basis.**