Homework 12

Jiahao Liu April 20, 2024

1. Max Equal Cut Problem

Given an undirected graph G(V, E), where V has an even number of vertices, find an **equal partition of V into two sets A and B, maximizing the size of E(A, B)**, where E(A, B) denote the set of edges with one endpoint in A and one endpoint in B. Provide a factor 2-approximation algorithm for solving the Max Equal Cut problem and prove the approximation ratio for the algorithm.
Iteratively build A and B from empty. At each step consider a pair of vertices u, v in $V \backslash (A \cup B)$, count the edges between u, v and nodes in A, B. If $|E_{uA}| + |E_{vB}| >= |E_{uB}| + |E_{vA}|$, add u to B and v to A. Otherwise, add u to A and v to B. Repeat this process until all vertices are assigned to A or B.

Proof. Suppose the algorithm produces A and B and the optimal solution is $A^* and\ B^*$, then we can count the number of edges in $E_A, E_B, E(A, B)$ in the following way: let $(a_1, b_1), (a_2, b_2), \ldots, (a_{|V|/2}, b_{|V|/2})$ be the order of vertices being added into A, B by our algorithm. Then for any $i \in [1, |V|/2]$, we have $|E_{a_iA}| + |E_{b_iB}| \leq |E_{a_iB}| + |E_{b_iA}|$. The left hand side of the inequality is just the number of edges a_i, b_i contribute to E_A, E_B, while the right hand side is the number of edges a_i, b_i contribute to $E(A, B)$. Adding up all the inequalities, we have $E_A + E_B \leq E(A, B)$. Since $E_A + E_B + E(A, B) = |E|$, we have $E(A, B) \geq |E|/2 \geq E(A^*, B^*)/2$, so $E(A^*, B^*) \leq 2 * E(A, B)$ □

2. Integer LP with Natural Number Variables

650 students in the Analysis of Algorithms class in 2024 Spring take the exams onsite. The university provided 7 classrooms for exam use, each classroom i can contain C_i (capacity) students. The safety level of a classroom is defined as $\alpha_i(C_i - S_i)$, where α_i is the known parameter for classroom i, and S_i is the actual number of students placed to take the exams in the classroom. We want to maximize the total safety level of all the classrooms. Besides, to guarantee good spacing, the number of students in a classroom should not exceed half of the capacity of each classroom.
Express the problem as a integer linear programming problem to obtain the number of students to be placed in each room. You DO NOT need to numerically solve the program.
Variables: x_i is the number of students placed in classroom i.
Objective Function: Maximize $\sum_{i=1}^{7} \alpha_i(C_i - x_i)$
Constraints: $\forall i \in [1, 7], x_i \in \mathbb{Z}^+,\ 0 \leq x_i \leq S_i/2 \quad \sum_{i=1}^{7} x_i \leq 650 \quad \sum_{i=1}^{7} x_i \geq 650$

3. Min-S-T-Cut Problem

Given a directed network G = (V, E) with source s and sink t, the capacity of each edge is given. The goal is to find a minimum s-t cut in the network. Write down this problem as an Integer Linear Program.
Variables: $x_i = \{0, 1\}$ is the set indicator for node $i \in V$, $x_i = 1 \Rightarrow i \in A, x_i = 0 \Rightarrow i \in B$
Nonlinear Objective Function: Suppose the capacity of directed edge $e = (i, j)$ is c_{ij}, find the minimum cut C(A, B):

$$\min\{\sum_{(i,j)\in E} c_{ij} x_i (1 - x_j)\}$$

Constraints: $\forall i \in V \backslash \{s, t\}, x_i = \{0, 1\} \quad x_s = 1 \quad x_t = 0$
Variables: $x_i = \{0, 1\}$ is the set indicator for node $i \in V$, $x_i = 1 \Rightarrow i \in A, x_i = 0 \Rightarrow i \in B$, $y_{ij} = \{0, 1\}$ is the set indicator for edge $(i, j) \in E$, $y_{ij} = 1 \Rightarrow (i, j) \in Cut$
Linear Objective Function:

$$\min\{\sum_{(i,j)\in E} c_{ij} y_{ij}\}$$

Constraints: $\forall (i, j) \in E, y_{ij} \geq x_i - x_j \quad \forall (i, j) \in E, y_{ij} = \{0, 1\} \quad \forall i \in V \backslash \{s, t\}, x_i = \{0, 1\} \quad x_s = 1 \quad x_t = 0$
The first constraint is to ensure that when $x_i = 1, x_j = 0$, then $y_{ij} = 1$, which means edge (i, j) is in the cut.
Note that first constraint does not directly constrain the value of y_{ij} if $(x_i, x_j) = (0, 1), (1, 1), (0, 0)$, while minimized the total weight will try to set $y_{ij} = 0$ whenever possible.
The last constraint is to ensure that we get a valid cut as the solution. Otherwise, the LP-solver would try to put all the variables on the same side so that no edges need to be included, which can be avoided by forcing s and t to be on opposite sides.

4. Dividable Knapsack Problem

Suppose you have a knapsack with maximum weight W and maximum volume V. We have n dividable objects. Each object i has value m_i, weights w_i and takes v_i volume. Now we want to maximize the total value in this knapsack, and at the same time we want to use up all the space (volume) in the knapsack. Formulate this problem as a linear programming problem. You DO NOT have to solve the resulting LP.
Variables: x_i is the proportion of object i to be included into the knapsack.
Objective Function: Maximize $\sum_{i=1}^{n} m_i x_i$
Constraints: $\forall i \in [1, n], 0 \le x_i \le 1 \quad \sum_{i=1}^{n} w_i x_i \le W \quad \sum_{i=1}^{n} v_i x_i \le V \quad \sum_{i=1}^{n} v_i x_i \ge V$

5. Max Cut Problem

A Max-Cut of an undirected graph G = (V,E) is defined as a cut $C_{\max}$ such that the number of edges crossing $C_{\max}$ is the maximum possible among all cuts. Consider the following algorithm.
i) Start with an arbitrary cut C.
ii) While there exists a vertex v such that moving v from one side of C to the other increases the number of edges crossing C, move v and update C.
Prove that the algorithm is a 2-approximation, that is the number of edges crossing $C_{\max}$ is at most twice the number crossing C.
Similar prove to Q1, suppose above algorithm end with cut $C = (A, B)$, then for each node $a_i \in A$, we have $|E_{a_i}| \le |E_{a_i B}|, E_{a_i B} \subset E(A, B)$; For each node $b_j \in B$, we have $|E_{b_j}| \le |E_{b_j A}|, E_{b_j A} \subset E(A, B)$. Therefore, each node contributes edges to E_A, E_B at most the same with its edge contribution to $E(A, B)$. Adding up the contribution edges of all nodes, we have $E_A + E_B \le E(A, B)$. Since $|E_A| + |E_B| + |E(A, B)| = |E|$, we have $C = |E(A, B)| \ge |E|/2 \ge C_{\max}/2$, so $C_{\max} \le 2 * C$.