School of Computer Science and Technology
University of Science and Technology of China

Exercise Sheet 4 for Design and Analysis of Algorithms Autumn 2022

Solution

Exercise 1 (30 points, graded by Di Wu)
Let G be a complete undirected graph in which all edge lengths are either 1 or 2 (clearly, G satisfies the triangle inequality). Give a 4/3-approximation algorithm for TSP in this special class of graphs.

Hint: Start by finding a minimum 2-matching in G. A 2-matching is a subset S of edges so that every vertex has exactly 2 edges of S incident at it. You can assume that finding such a minimum 2-matching can be done in polynomial time.

Solution. The algorithm FINDTOUR-3 is given as follows:

(1) Find a minimum 2-matching in G, which consists of one or several components, denoted as $C = \{C_1, C_2, \ldots, C_k\}$.

(2) Convert to TSP: Choose an arbitrary edge (u_i, v_i) in each component C_i $(1 \leq i \leq k)$. Remove all such edges in all the components and connect v_j to u_{j+1} for all $1 \leq j \leq k-1$. Moreover, connect v_k to u_1. (See Figure 1)

(3) Output the result of step (2), which is a TSP tour, denoted as $\mathcal{T}$. (Note: The starting vertex can be arbitrary)

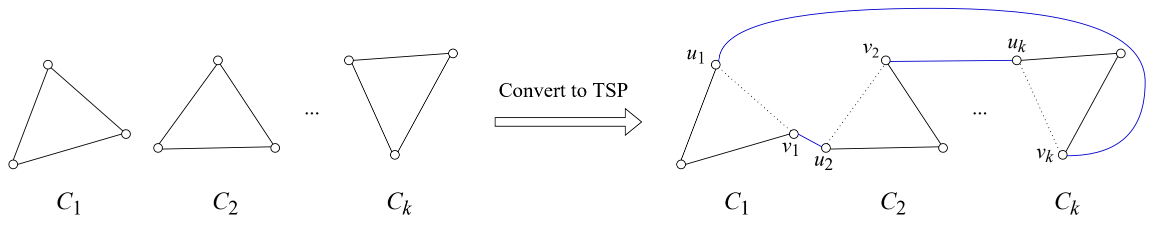

Figure 1: An illustration of step (2)

Analysis of the algorithm:

(a) Running time: Finding a minimum 2-matching and other steps all run in polynomial time.

(b) Correctness: Recall that a TSP tour is a cycle which visits each vertex in G exactly once. Therefore, this tour enters and exits each vertex via different edges. Hence, a TSP tour in G is also a 2-matching for G. Hence, the cost of the minimum 2-matching is a lower bound for the cost of any TSP tour. Hence, $\text{cost}(C) \leq \text{OPT}$, where OPT is the cost of an optimal TSP tour.

In each component C_i of the 2-matching, each vertex is incident to exactly two edges. Due to this property, each of these components consists of at least 3 vertices (If a component has less than three vertices then each vertex cannot be incident to exactly 2 edges). Therefore, there exist at most $\frac{n}{3}$ such components, i.e., $k \leq \frac{n}{3}$, where n is the number of vertices in G.

In step (2) we remove k edges and add k edges to the minimum 2-matching. By considering the worst case that all the k removed edges have length 1 and all the added edges have length 2, the length of the TSP tour output by the algorithm is

$$\begin{aligned}
\operatorname{cost}(\mathcal{T}) &\le \operatorname{cost}(C) - k + 2k \\
&= \operatorname{cost}(C) + k \\
&\le \operatorname{cost}(C) + \frac{n}{3} \\
&\le \operatorname{cost}(C) + \frac{\operatorname{cost}(C)}{3} \\
&= \frac{4}{3}\operatorname{cost}(C) \\
&\le \frac{4}{3}\mathrm{OPT},
\end{aligned}$$

where the third inequality follows from the fact that $\operatorname{cost}(C) \ge n$ since the minimum 2-matching consists of n edges each of length at least 1.

Therefore, FindTour-3 is a 4/3-approximation algorithm for TSP in this special class of graphs.

Exercise 2 **(30 points, graded by Yinhao Dong)**
Consider the following modification to the metric uncapacitated facility location problem. Define the cost of connecting client j to facility i to be c_{ij}^2. The c_{ij}'s still satisfy the triangle inequality (but the new connection costs, of c_{ij}^2, do not). Show that the algorithm LP-FacilityLocation in Lecture 10 achieves a constant approximation ratio for this case.

Proof. Recall that in the first step of the algorithm LP-FacilityLocation, we compute an optimal solutions (x^*, y^*) and (v^*, w^*) to the primal and dual LP. By complementary slackness, $x_{ij}^* > 0$ implies $v_j^* - w_{ij}^* = c_{ij}^2$ and $c_{ij}^2 \le v_j^*$ after the modification.
For this case, the service cost for client j in cluster k is

$$\begin{aligned}
c_{i_k j}^2 &\le (c_{ij} + c_{ij_k} + c_{i_k j_k})^2 && \text{(by triangle inequality)} \\
&\le 3\left(c_{ij}^2 + c_{ij_k}^2 + c_{i_k j_k}^2\right) && \text{(by Cauchy-Schwarz inequality)} \\
&\le 3\left(v_j^* + 2v_{j_k}^*\right) && \text{(by complementary slackness)} \\
&\le 3 \cdot 3v_j^* && (v_{j_k}^* \le v_j^* \text{ according to the algorithm}) \\
&= 9v_j^*,
\end{aligned}$$

where i is the common neighbor of j and j_k.
Hence the total service cost is at most $9\sum_{j\in\mathcal{D}} v_j^*$. Note that $\sum_{j\in\mathcal{D}} v_j^* = \mathrm{OPT}_{\text{dual LP}} = \mathrm{OPT}_{\text{LP}}$, where $\mathrm{OPT}_{\text{dual LP}}$ is the value of the optimum solution of the dual LP, and OPT_{LP} is the value of the optimum solution of the corresponding primal LP.
According to the analysis in Lecture 10, the facility cost f_{i_k} can be bounded by $f_{i_k} \le \sum_{i:\text{ neighbor of } j_k} f_i y_i^*$.
Since $j_k, j_{k'}$ cannot have a common neighbor for $k \ne k'$, the total facility cost is $\sum_k f_{i_k} \le \sum_{i\in\mathcal{F}} f_i y_i^* \le \sum_{i\in\mathcal{F}} f_i y_i^* + \sum_{i\in\mathcal{F}}\sum_{j\in\mathcal{D}} c_{ij}^2 x_{ij}^* = \mathrm{OPT}_{\text{LP}}$.
Since $\mathrm{OPT}_{\text{LP}} \le \mathrm{OPT}$, where OPT is the value of the optimum solution for the metric UFL problem, the total cost is at most

$$9\sum_{j\in\mathcal{D}} v_j^* + \sum_{i\in\mathcal{F}} f_i y_i^* \le 10 \cdot \mathrm{OPT}_{\text{LP}} \le 10 \cdot \mathrm{OPT}.$$

□

Exercise 3 **(40 points, graded by Yudong Zhang)**
Consider the MAX 2SAT problem, in which every clause has at most two literals. As in the case of the maximum cut problem, we'd like to express the MAX 2SAT problem as an SDP, and then round its optimal solution to get an approximate solution to the MAX 2SAT problem. Derive a 0.878-approximation algorithm for the MAX 2SAT problem.

Solution. Corresponding to each boolean variable x_i $(1 \le i \le n)$, introduce variable $y_i \in \{-1,1\}$. In addition, introduce another variable $y_0 \in \{-1,1\}$. Let us impose the convention that x_i is TRUE iff $y_i = y_0$ and FALSE otherwise.
Under this convention we can write the value of a clause in terms of the y_i's, where the *value*, $v(C)$, of clause C is defined to be 1 if C is satisfied and 0 otherwise. Thus, for clauses containing only one literal, we have

$$\begin{aligned} v(x_i) &= \frac{1+y_0y_i}{2} \\ v(\neg x_i) &= \frac{1-y_0y_i}{2} \end{aligned}$$

For clauses containing 2 literals, we have

$$\begin{aligned} v(x_i \vee x_j) &= 1 - v(\neg x_i)\, v(\neg x_j) \\ &= 1 - \frac{1-y_0y_i}{2} \cdot \frac{1-y_0y_j}{2} \\ &= \frac{3 + y_0y_i + y_0y_j - y_0^2 y_i y_j}{4} \\ &= \frac{1+y_0y_i}{4} + \frac{1+y_0y_j}{4} + \frac{1-y_iy_j}{4} \\ v(\neg x_i \vee x_j) &= \frac{1-y_0y_i}{4} + \frac{1+y_0y_j}{4} + \frac{1+y_iy_j}{4} \\ v(x_i \vee \neg x_j) &= \frac{1+y_0y_i}{4} + \frac{1-y_0y_j}{4} + \frac{1+y_iy_j}{4} \\ v(\neg x_i \vee \neg x_j) &= \frac{1-y_0y_i}{4} + \frac{1-y_0y_j}{4} + \frac{1-y_iy_j}{4} \end{aligned}$$

It is easy to check that the value of a 2 literal clause consists of a linear combination of terms of the form $(1+y_iy_j)$ or $(1-y_iy_j)$, so by collecting the coefficients of like terms we can write the objective function as follows, where the values a_{ij}, b_{ij} are constants ≥ 0:

$$\begin{array}{lll} \max & \sum_{0 \le i < j \le n} \left[a_{ij}(1+y_iy_j) + b_{ij}(1-y_iy_j)\right] & \\ \text{s.t.} & y_i \in \{-1,1\} & 0 \le i \le n \end{array}$$

We set up the SDP as follows, where vector variable $\boldsymbol{v}_i$ corresponds to y_i:

$$\begin{array}{lll} \max & \sum_{0 \le i < j \le n} \left[a_{ij}(1+\langle \boldsymbol{v}_i, \boldsymbol{v}_j\rangle) + b_{ij}(1-\langle \boldsymbol{v}_i, \boldsymbol{v}_j\rangle)\right] & \\ \text{s.t.} & \langle \boldsymbol{v}_i, \boldsymbol{v}_i\rangle = 1 & 0 \le i \le n \\ & \boldsymbol{v}_i \in \mathbb{R}^{n+1} & 0 \le i \le n \end{array}$$

As in the case of the maximum cut problem, we give an approximation algorithm SDP-MAX-2SAT for the MAX 2SAT problem as follows:

(1) Solve the SDP for MAX 2SAT to obtain the optimum solution $\boldsymbol{v}_i, 0 \le i \le n$.

(2) Choose a uniformly random unit vector $\boldsymbol{r} \in \mathbb{R}^{n+1}$.

(3) Set values y_i for $0 \le i \le n$ by

$$y_i = \begin{cases} 1 & \text{if } \langle \boldsymbol{r}, \boldsymbol{v}_i\rangle \ge 0 \\ -1 & \text{otherwise} \end{cases}$$

(4) For each $1 \le i \le n$, set

$$x_i = \begin{cases} \text{TRUE} & \text{if } y_i = y_0 \\ \text{FALSE} & \text{otherwise} \end{cases}$$

Analysis of the algorithm:

(a) Running time: Solving SDP and other steps all run in polynomial time.

(b) Correctness: Let W be the random variable denoting the value of satisfied clauses. We have $W = \sum_{0\leq i<j\leq n} [a_{ij}(1+y_iy_j) + b_{ij}(1-y_iy_j)]$. Let $\theta_{ij} \in [0,\pi]$ be the angle between $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$. From the analysis for SDP-MaxCut in Lecture 11, we know $\Pr[y_iy_j = -1] = \frac{2\theta_{ij}}{2\pi} = \frac{\theta_{ij}}{\pi} \geq 0.878 \cdot \frac{1-\cos\theta_{ij}}{2}$. This also shows $\Pr[y_iy_j = 1] = 1 - \frac{\theta_{ij}}{\pi} \geq 0.878 \cdot \frac{1+\cos\theta_{ij}}{2}$. Therefore,

$$\mathrm{E}[y_iy_j] = 1 \cdot \Pr[y_iy_j = 1] + (-1) \cdot \Pr[y_iy_j = -1] = 1 \cdot \left(1 - \frac{\theta_{ij}}{\pi}\right) + (-1) \cdot \frac{\theta_{ij}}{\pi} = 1 - \frac{2\theta_{ij}}{\pi}$$

By linearity of expectation, we have

$$\begin{aligned}
\mathrm{E}[W] &= \mathrm{E}\left[\sum_{0\leq i<j\leq n} [a_{ij}(1+y_iy_j) + b_{ij}(1-y_iy_j)]\right] \\
&= \sum_{0\leq i<j\leq n} [a_{ij}(1+\mathrm{E}[y_iy_j]) + b_{ij}(1-\mathrm{E}[y_iy_j])] \\
&= \sum_{0\leq i<j\leq n} \left[a_{ij}\left(2 - \frac{2\theta_{ij}}{\pi}\right) + b_{ij} \cdot \frac{2\theta_{ij}}{\pi}\right] \\
&= 2\sum_{0\leq i<j\leq n} \left[a_{ij}\left(1 - \frac{\theta_{ij}}{\pi}\right) + b_{ij} \cdot \frac{\theta_{ij}}{\pi}\right] \\
&\geq 0.878 \sum_{0\leq i<j\leq n} [a_{ij}(1+\cos\theta_{ij}) + b_{ij}(1-\cos\theta_{ij})] \\
&= 0.878 \sum_{0\leq i<j\leq n} [a_{ij}(1+\langle \boldsymbol{v}_i, \boldsymbol{v}_j\rangle) + b_{ij}(1-\langle \boldsymbol{v}_i, \boldsymbol{v}_j\rangle)] \\
&= 0.878 \cdot \mathrm{OPT}_{\mathrm{SDP}} \\
&\geq 0.878 \cdot \mathrm{OPT},
\end{aligned}$$

where $\mathrm{OPT}_{\mathrm{SDP}}$ is the value of the optimum solution of SDP, and OPT is the value of the optimum solution of the MAX 2SAT problem.

Therefore, SDP-MAX-2SAT is a 0.878-approximation algorithm for the MAX 2SAT problem.
