CS880: Approximation and Online Algorithms	**Scribe:** Ashwin Maran
Lecture 5: Random Sampling; Single-source rent-or-buy	**Date:** 9/25/2019

5.1 Recap

Consider the 2D Euclidean TSP we saw in Lecture 3. The problem statement is as follows.

Definition 5.1.1 (2D Euclidean TSP)
Given n points on a plane, find a tour of the shortest Euclidean length.

In Lecture 3, we made the following four assumptions about the problem and the optimal tour.

1. The n points lie on $n^2 \times n^2$ grid.
2. The tour crosses the boundaries of squares only at 'portals'.
3. The tour is non-crossing.
4. The tour crosses each portal at most twice.

We showed that Assumption 1 can be made with at most a loss of $(1 + 1/n)$ factor in the approximation ratio. We also showed that we can make Assumptions 3 and 4 without any incurring any loss in the approximation ratio.

We then considered the following algorithm to solve the 2D Euclidean TSP.

Algorithm 1 (2D Euclidean TSP)

Given: n points on a $n^2 \times n^2$ grid.

1: Divide the grid into four sub-grids.
2: Place m equidistant portals on each side of the boundary of each sub-grid.
3: Within each subgrid recursively solve for collection of line segments that enter and exit through portals and capture all the points within the grid.

We showed that Algorithm 1 runs in $poly(n) \cdot 2^{O(m)}$ time. We also found in Lecture 3 that in the worst case, Assumption 2 leads to a huge loss in the approximation ratio. In this lecture, we will overcome this limitation by making use of randomization.

5.2 PTAS for the Euclidean TSP

In Algorithm 1, since the grid is recursively subdivided, eventually every gridline will divide some subgrid. We will refer to the gridline that divides the grid at the i^{th} iteration of the algorithm, to be a 'level i gridline'. Notice that a level i gridline has $2^i \cdot m$ portals on it.

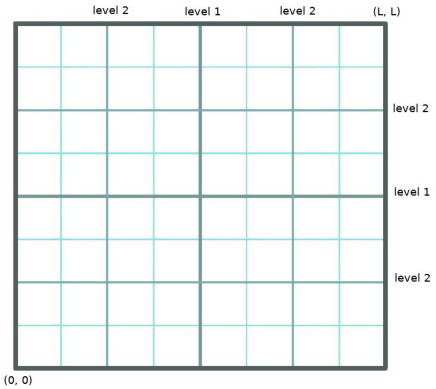

Figure 5.2.1: $n^2 \times n^2$ grid with gridlines of different levels.

Let $L = n^2$ be the size of the original grid. Therefore, the distance between any two portals on a level i gridline is $L/(2^i \cdot m)$. This implies that every time a tour crosses a level i line, because of Assumption 2, the length of the tour increases by at most $L/(2^i \cdot m)$.
For any gridline l at the i^{th} level, let t_l be the number of times the tour crosses the line l.

$$\therefore \text{Increase in length due to Assumption 2} \leq \sum_{\text{levels } i} \left(\sum_{\text{lines } l} t_l \cdot \frac{L}{2^i \cdot m} \right) \tag{5.2.1}$$

The problem with Algorithm 1 is that t_l could be high for lines at a low level. This would result in a large increase in the length of the tour. Before we fix this issue using randomization, consider the following lemma.

Lemma 5.2.1 $\sum_l t_l \leq 4 \cdot OPT$

Proof: Fix any segment of the OPT tour. Let it be x units in the $x-$direction and y units in the $y-$direction.

$$\therefore \text{Euclidean length} = \sqrt{x^2 + y^2}$$

The number of crossings is equal to the sum of horizontal and vertical gridlines that the line segment intersects. The number of horizontal gridlines that the line segment intersects would be $x+1$, if x is an integer and the line segment starts at a point on the grid, and it would be x in all other cases. Similarly, the number of vertical gridlines intersected would be $y+1$ if y is an integer and the line segment starts at a point on the grid, and it would be y in all other cases.

The Cauchy-Schwarz inequality implies that

$$x + y \leq \sqrt{2} \left(\sqrt{x^2 + y^2} \right) \ \forall \ x, y \geq 0$$

In the event that the number of horizontal gridlines that are intersected is $x + 1$, the following inequality holds since it is only possible when x is a positive integer.

$$\therefore x + y + 1 \leq \left(\sqrt{2} + 1 \right) \left(\sqrt{x^2 + y^2} \right)$$

The same applies for the event when the number of vertical gridlines that are intersected is $y+1$. When both the horizontal and vertical gridlines that are intersected are $x+1$ and $y+1$ respectively,

$$x + y + 2 \leq \left(2\sqrt{2}\right)\left(\sqrt{x^2+y^2}\right)$$

Therefore in any event

$$\text{Number of crossings by the line segment} \leq 4\left(\sqrt{x^2+y^2}\right)$$

If we sum over all the line segments in the OPT tour, we find that

$$\sum_l t_l \leq 4 \cdot OPT \tag{5.2.2}$$

■

Lemma 5.2.1 tells us that $\sum_l t_l$ is bounded above. Therefore if for some l, the value of t_l is really high, the value of t_l must be low for the other gridlines. We will use this fact to devise a randomized algorithm that overcomes the limitations of Algorithm 1.

Algorithm 2 (2D Euclidean TSP)

Given: n points on a $L \times L$ grid.

1: Create a $\{-L, L\} \times \{-L, L\}$ grid.
2: Move the origin of the given $L \times L$ grid to a point within the region $\{-L/2, L/2\} \times \{-L/2, L/2\}$, uniformly at random.
3: Apply Algorithm 1 on the $\{-L, L\} \times \{-L, L\}$ grid.

Figure 5.2.2: Algorithm 2

Note that for any given gridline l in Algorithm 2, and for any level i,

$$\Pr\left[l \text{ is a level } i \text{ gridline}\right] = \frac{2^i}{L} \tag{5.2.3}$$

$\therefore$ from Equation (5.2.1),

$$\begin{aligned}
\mathbb{E}\left[\text{Length of detours}\right] &\leq \sum_{\text{levels } i}\left(\sum_{\text{lines } l} \Pr\left[l \text{ is a level } i \text{ gridline}\right] \cdot t_l \cdot \frac{L}{2^i \cdot m}\right) \\
&= \sum_{\text{levels } i}\left(\sum_{\text{lines } l}\left(\frac{2^i}{L}\right) \cdot t_l \cdot \frac{L}{2^i \cdot m}\right) \\
&= \frac{1}{m}\sum_{\text{levels } i}\left(\sum_{\text{lines } l} t_l\right) \\
&\leq \frac{1}{m}\sum_{\text{levels } i}\left(4 \cdot OPT\right)
\end{aligned}$$

$$\therefore \mathbb{E}\left[\text{Length of detours}\right] \leq \frac{12 \cdot OPT}{m} \cdot \log n \tag{5.2.4}$$

So, if we set $m = \frac{12}{\varepsilon} \log n$, then

$$\mathbb{E}[\text{Length of detours}] \leq \varepsilon \cdot OPT$$

Therefore, when we use Algorithm 2, in expectation Assumption 2 leads to a loss in the approximation factor of at most $(1 + \varepsilon)$. By standard techniques of boosting, we can claim that with a high probability, Assumption 2 will lead to a loss in approximation factor of at most $(1 + \varepsilon)$.

Moreover, just like Algorithm 1, Algorithm 2 also runs in $poly(n) \cdot 2^{O(m)} = poly(n) \cdot 2^{1/\varepsilon}$ time. Therefore, there exists a randomized PTAS for the 2D Euclidean TSP Problem.

5.3 Buy-at-bulk network design problem

In the general buy-at-bulk network design problem, we are given a graph $G = (V, E)$ with edge lengths c_e, and set of k commodities with source-sink pairs $(s_1, t_1), ..., (s_k, t_k)$ and demands, $d_1, ..., d_k$. The cost of buying a capacity of x on any edge per unit length is given by the function $f(x)$, which is a sub-additive function (i.e., $f(x_1) + f(x_2) \geq f(x_1 + x_2)$). We are to buy capacities of each edge, such that d_i of each commodity can flow from s_i to t_i using the purchased capacities. The goal of the problem is to minimize the cost of purchasing these capacities.

Let us consider the case where there is a single source s and k source-sink pairs (s, t_i) for all $t_i \in T \subset V$, each with a demand of 1.

If the cost function f is a linear function (i.e., $f(x) = a \cdot x$), then the problem becomes an instance of the shortest-path problem. If the cost function f is a constant function (i. e., $f(x) = c \cdot \mathbf{1}_{x \neq 0}$), then the problem becomes an instance of the Steiner-tree problem. We know that the shortest-path problem can be solved exactly in polynomial time, and a 2-approximation for the Steiner tree problem can be found in polynomial time as well.

The problem gets more interesting when the cost function f is linear up to a point, and constant afterwards (i. e., $f(x) = \min\{a \cdot x, M\}$). This problem is also called a single-source rent-or-buy problem. We can understand the function f to mean that it is cheaper to rent an edge (at a cost of $a \cdot x$) until a certain capacity(M), after which it is cheaper to buy the edge.

5.4 Single-source rent-or-buy problem

The single-source rent-or-buy problem is formally defined as follows:

Definition 5.4.1 *Given a graph $G = (V, E)$ with edge lengths c_e, a source s and a terminal set $T \subset V$, and a constant $M > 0$. The single-source rent-or-buy problem is the problem of finding*

paths P_t from s to every $t \in T$, such that the following quantity is minimized.

$$\sum_{e \in \bigcup_t P_t} \min\left\{(M \cdot c_e), (|\{t : P_t \ni e\}| \cdot c_e)\right\}$$

Notice that if we can somehow identify the edges that need to be bought, then the single-source rent-or-buy problem can be broken down into the shortest-path and Steiner tree sub-problems. The following algorithm uses randomization to do the same.

Algorithm 3 (Single-source rent-or-buy)

Given : Graph $G = (V, E)$, edge lengths c_e, $s \in V$, $T \subset V$, and $M > 0$

1: For each $t \in T$, select it independently with probability $1/M$ and place it in a set B (Buy set).
2: Construct a 2-approximate Steiner tree over $\{s\} \bigcup B$, called F. Buy the edges in F.
3: Connect every remaining terminal $t \in T \backslash B$ to F via the shortest paths. Rent these paths.

Consider the following lemmas.

Lemma 5.4.2

$$\mathbb{E}[M \cdot c(F)] \leq 2 \cdot OPT \tag{5.4.5}$$

where $c(F) = \sum_{e \in F} c_e$

Lemma 5.4.3

$$\mathbb{E}[\textit{rent cost}] \leq \mathbb{E}[\textit{buy cost}] \tag{5.4.6}$$

Let us assume that the two lemmas are true, for now.

Theorem 5.4.4 *Algorithm 3 is a $1/4$-approximation algorithm for the single-source rent-or-buy problem.*

Proof:

$$\begin{aligned}
\mathbb{E}[\text{cost of ALGO}] &= \mathbb{E}[\text{rent cost}] + \mathbb{E}[\text{buy cost}] \\
&\leq 2 \cdot \mathbb{E}[\text{buy cost}] \\
&= 2 \cdot \mathbb{E}[M \cdot c(F)] \\
&\leq 4 \cdot OPT \\
\therefore \frac{OPT}{\mathbb{E}[\text{cost of ALGO}]} &\geq \frac{1}{4}
\end{aligned} \tag{5.4.7}$$

■

We will now prove Lemma 5.4.2.

Proof: We will prove the lemma by showing that in expectation, purchasing the optimal Steiner tree over $\{s\} \cup B$ costs less than OPT. That would imply the statement of the lemma, since we are using a 2-approximation algorithm to find the Steiner tree, F.

We will show that the optimal Steiner tree has cost less than OPT by demonstrating the existence of a Steiner tree spanning $\{s\} \cup B$ which has an expected cost less than OPT.

Consider the optimal solution to the single-source rent-or-buy problem. Consider the sub-tree of the optimal solution, that spans $\{s\} \cup B$. We will denote it by F^* The expected cost of purchasing this tree is

$$\mathbb{E}[M \cdot c(F^*)] = \sum_{e \in OPT} M \cdot c_e \cdot \Pr[e \in F^*]$$

Now, note that

$$\Pr[e \in F^*] \leq \frac{1}{M} \cdot \#\text{terminals in OPT using } e$$

Also, it is trivially true that $\Pr[e \in F^*] \leq 1$.

$$\therefore \Pr[e \in F^*] \leq \min\left\{1, \frac{1}{M} \cdot \#\text{terminals in OPT using } e\right\}$$

$$\begin{aligned}
\therefore \mathbb{E}[M \cdot c(F^*)] &\leq \sum_{e \in OPT} \min\left\{1, \frac{1}{M} \cdot \#\text{terminals in OPT using } e\right\} \cdot (M \cdot c_e) \\
&= \sum_{e \in OPT} \min\left\{M \cdot c_e, (\#\text{terminals in OPT using } e) \cdot c_e\right\} \\
&= OPT
\end{aligned}$$

$$\therefore \mathbb{E}[M \cdot c(F)] \leq 2 \cdot OPT$$

■

The proof of Lemma 5.4.3 will be provided in Lecture 6.