CS880: Approximation and Online Algorithms	**Scribe:** Dongqiangzi Ye
Lecture 3: Poly Time Approximation Schemes: Knapsack; Euclidean TSP	**Date:** 09/09/2019

3.1 Polynomial Time Approximation Scheme (PTAS)

Definition 3.1.1 *PTAS: An algorithm that for any given constant $\epsilon > 0$, produce a $(1+\epsilon)$ approximation with running time polynomial in size of the instant.*

Definition 3.1.2 *Fully PTAS (FPTAS): An algorithm that for any constant $\epsilon > 0$, returns a $(1+\epsilon)$ approximation with running time polynomial in both the instance size n and $\frac{1}{\epsilon}$.*

3.2 Knapsack

Definition 3.2.1 *Knapsack: Given n items with size s_i and value v_i, and a knapsack of size B, find subset $S \subseteq [n]$ with $\sum\limits_{i \in S} s_i \leq B$ that maximizes $\sum\limits_{i \in S} v_i$.*

Assumption $V = \max v_i$. All v_i's are integers in $\{1, ..., V\}$

Claim 3.2.2 *There is a DP that solves knapsack exactly in pseudo polynomial time $O(n^2V)$*

Algorithm 1 DP for Knapsack

$A(1) \longleftarrow \{(0,0),(s_1,w_1)\}$
for $j = 2..n$ **do**
  $A(j) \longleftarrow A(j-1)$
  for each $(t,w) \in A(j-1)$ **do**
    if $t + s_j \leq B$ **then**
      Add $(t+s_j, w+v_j)$ to $A(j))$
    end if
  end for
  Remove dominated pairs from $A(j)$
end for
return $\max_{(t,w)\in A(n)} w$

Claim 3.2.3 *We can design a FPTAS that solves Knapsack in time $O(\frac{n^3}{\epsilon})$.*

Algorithm 2 Discretization DP method for Knapsack

Define $V = \max v_i$, $R = (1+\frac{1}{\epsilon})n$
Round value down to integers in $\{1, ..., R\}$, $v_i' = \lfloor v_i \frac{R}{V} \rfloor$
Run Algorithm 1 to solve instance exactly over v_i'
return Solution

Running Time $= O(n^2R) = O(\frac{n^3}{\epsilon}) \longrightarrow$ FPTAS

Proof: At first, we define some notations:

$$
\begin{aligned}
OPT &= \text{optimal value over } v_i\text{'s} \\
OPT^{'} &= \text{optimal value over } v_i^{'}\text{'s} \\
ALG &= \text{algorithm's value over } v_i\text{'s}
\end{aligned}
$$

Then we want to define R such that:

$$
\frac{R}{V}OPT - \frac{1}{1+\epsilon}\frac{R}{V}OPT \geq n \tag{3.2.1}
$$

$$
\frac{\epsilon}{1+\epsilon}\frac{R}{V}OPT \geq n
$$

$$
R \geq \frac{1+\epsilon}{\epsilon}\frac{Vn}{OPT}
$$

Since one possible solution is to put the most valuable item in a knapsack by itself, $OPT \geq V$. Then we can pick:

$$
R = (1 + \frac{1}{\epsilon})n \tag{3.2.2}
$$

By (3.2.1) and (3.2.2), we can infer this statements:

$$
\begin{aligned}
OPT^{'} &\geq \sum_{i \in OPT} v_i^{'} \\
&\geq \sum_{i \in OPT} \lfloor v_i \frac{R}{V} \rfloor \\
&\geq \sum_{i \in OPT} (v_i \frac{R}{V} - 1) \\
&= \frac{R}{V}OPT - n \\
&\overset{(3.2.1)}{\geq} \frac{1}{1+\epsilon}\frac{R}{V}OPT
\end{aligned}
$$

Since we round down values to get the $OPT^{'}$ and the algorithm's solution is the same subset which is still feasible, we have:

$$
\begin{aligned}
ALG &\geq OPT^{'}\frac{V}{R} \\
&\geq \frac{V}{R}\frac{1}{1+\epsilon}\frac{R}{V}OPT \qquad (3.2.3) \\
&= \frac{OPT}{1+\epsilon}
\end{aligned}
$$

Based on (3.2.2) and (3.2.3), this algorithm is FPTAS. ∎

3.3 2D Euclidean TSP

Definition 3.3.1 *Traveling Salesman Problem (TSP): Given n nodes and for each pair $\{i, j\}$ of distinct nodes, a distance $d_{i,j}$, we desire a closed path that visits each node exactly once (i.e., is a salesman tour) and incurs the least cost, which is the sum of the distances along the path.*

Definition 3.3.2 *2D TSP: the nodes lie in $\mathbb{R}^2$ (or more generally, in $\mathbb{R}^d$ for some d) and distance is defined using the ℓ_2 norm. i.e. figure 3.3.1, given n points on a plane, find a tour of the shortest Euclidean length.*

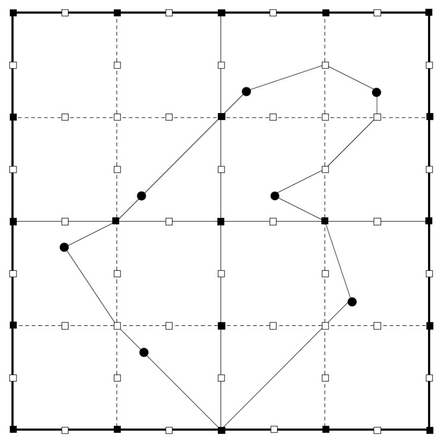

Figure 3.3.1: 2D Euclidean TSP

Definition 3.3.3 *Dissection: We will take a square of side length L around the points of the instance and divide it into four equally-sized square, then recursively divide each of these squares four equally-sized squares, and so on. e.g. Figure 3.3.2 left.*

Definition 3.3.4 *Portal: Tours that enter and exit the squares of the dissection happen at one of a small set of prespecified equidistant points (portals).*

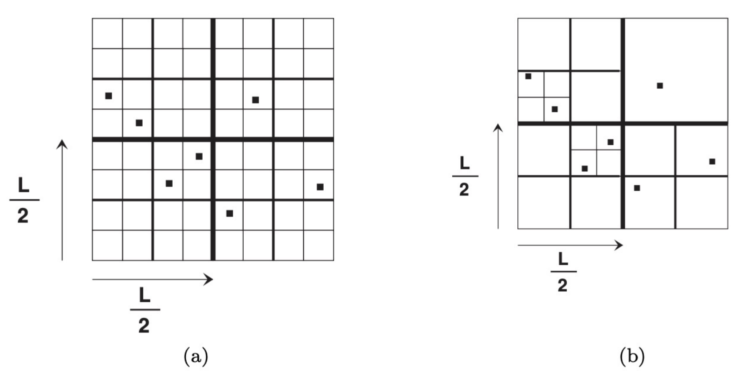

Figure 3.3.2: Left: The dissection. Right: The corresponding quadtree.

Assumption 1: Points lie on an integer grid. Points $\in [n^2]^2$.

Claim 3.3.5 *Discretization cost us at most $O(1 + \frac{1}{n})$ factor in the approximation radio.*

Proof: There are points on the boundary, so the lengh of the tour is at least n^2.

$$OPT \geq n^2$$

On the other hand, the cost of all detours $\leq \sqrt{n^2 + n^2} = \sqrt{2}n$ ■

Assumption 2: Tour must enter/exit a square at a portal.

Assumption 3: Optimal tour does not cross itself.

We create limits of enter/exist by assumption 3 without any loss of approximation factor. As we can see from Figure 3.3.3, if the tour cross itself, we can connect p1-p2 and p3-p4 to remove the crossing.

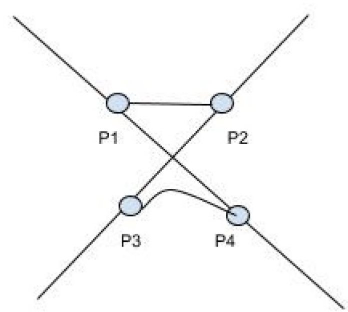

Figure 3.3.3: Illustration of Assumption 3

Assumption 4: Number of entries/exits at each portal is at most 1 each.

As we can see from Figure 3.3.4, if the tour cross three or more times at a portal, it can be shortcut to cross at most twice.

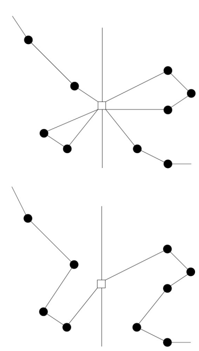

Figure 3.3.4: Illustration of Assumption 4

Algorithm 3 High Level Algorithm

Divide $n^2 \times n^2$ grid into four sub grids.
Divide the boundary of square into m equidistant portals on each side.
Within each subgrid recursively solve for collection of segments that capture all points.

Subproblem - box; for each portal number of entries and number of exits; matching of entries and exits.

For each portal number of entries and number of exits, we have 0/1 enter and 0/1 exit, so it's $O(4^{4m})$.

Matching of entries and exits is $O(2^{O(m)})$.

Number of different boxes in quadtree (Figure 3.3.2 right) is dominated by: $O(n^4)$.

Number of subproblems: $O(n^4 4^{4m} 2^{O(m)}) = O(n^4 2^{O(m)})$

So we can find the optimal solution satisfying assumptions 1-4 in time $poly(n)2^{O(m)}$.

References

[1] David P. Williamson, David B. Shmoys. The Design of Approximation Algorithms. 2010.

[2] Sanjeev Arora. Polynomial Time Approximation Schemes for Euclidean Traveling Salesman and other Geometric Problems. 1998.