SODA 24

A brief introduction to some interesting papers

Zhiwei Ying
2024-08-07

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

目录

1 **Other topics**

Shortest Disjoint Path on a Grid

2 **Online Algorithms**

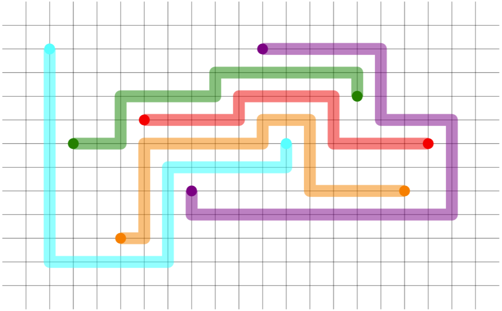
Shortest Disjoint Path on a Grid

Settings

Given a graph G on a grid, and $2k$ points $s_1, t_1, \ldots, s_k, t_k$. The goal is to find k vertex-disjoint paths that joins s_i, t_i for $i \in [k]$ with the total length of the solution to be minimized.

Settings

Given a graph G on a grid, and $2k$ points $s_1, t_1, \ldots, s_k, t_k$. The goal is to find k vertex-disjoint paths that joins s_i, t_i for $i \in [k]$ with the total length of the solution to be minimized.



Other Settings

1. The problem that given pairs of terminals with integer coordinates on a infinite grid, the player only need to decide whether vertex disjoint paths connecting these terminals exist is proved to be NP-Complete.

Other Settings

SHANGHAI JIAO TONG UNIVERSITY

1. The problem that given pairs of terminals with integer coordinates on a infinite grid, the player only need to decide whether vertex disjoint paths connecting these terminals exist is proved to be NP-Complete.
2. edge-disjoint
3. on directed graph or general planar graph
4. with length bounds on paths
5. Given a set of terminals, return the largest value k that there exist k vertex-disjoint paths joining a pair of terminals.

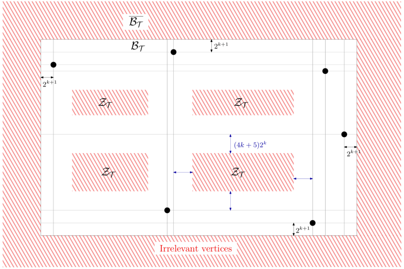
Result

Given a set $\Gamma = \{(s_1, t_1), \ldots, (s_k, t_k)\}$ of k pairs of terminals placed arbitrarily on a grid, we can find the shortest k vertex-disjoint paths between Γ or attest that there is no feasible solution in time $k^{2^{O(k)}} O([\Gamma])$, where $[\Gamma]$ is the number of bits required to encode Γ in binary.

Main Technique

Irrelevant vertices:

1. outside a bounding box of the terminals, whose border is at distance $\Theta(2^k)$ from the extreme terminals.
2. inside a bounding box of the terminals, whose border is at distance $\Theta(k \cdot 2^k)$ from the horizontal and vertical lines containing the terminals.



Main Technique

上海交通大学 SHANGHAI JIAO TONG UNIVERSITY

Irrelevant vertices:

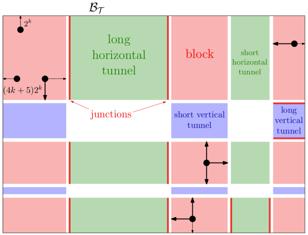
1. outside a bounding box of the terminals, whose border is at distance $\Theta(2^k)$ from the extreme terminals.
2. inside a bounding box of the terminals, whose border is at distance $\Theta(k \cdot 2^k)$ from the horizontal and vertical lines containing the terminals.

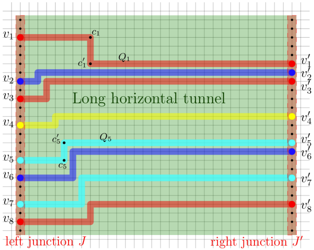
After excluding these irrelevant vertices, the number of solutions to enumerate is reduced to $\Theta(k \cdot 2^k \cdot L)$, where L is the distance between 2 terminals.

Structural lemma:
If there exists a routing of a set of terminals Γ, then there exists an optimal routing of Γ that does not use any irrelevant vertex.

Main Technique

Key lemma:
For those long horizontal tunnels that whose width is greater than its height, denote J_l and J_r as their junctions, and $\hat{J}_l = \{v_1, \ldots, v_t\} \subseteq J_l$ ordered from top to bottom, $\hat{J}_r = \{v'_1, \ldots, v'_t\} \subseteq J_r$ ordered from top to bottom. Then there exists a family $(Q_j)_{j=1}^t$ of pairwise disjoint paths.





Algorithm

上海交通大学 SHANGHAI JIAO TONG UNIVERSITY

Map V' that consists of all vertices in a block, in a short tunnel, or in a junction of a long tunnel to $\{0, 1, \ldots, k\}$. Enumerate all mappings and then get the result.

Running time:
The size of V' is $2^{O(k)}$. The number of all possible mappings is $(k+1)^{2^{O(k)}}$. All vertices in V' are contained in a bounding box that is at a distance at most 2^k from the coordinates of each vertex and can be encoded using $O(2(b + log(2^k))) = O([\Gamma])$ bits, where b is the maximum number of bits of a coordinate of a point in Γ. And we can decide in time poly($2^{O(k)}$) whether this configuration is valid or not.
Thus the overall running time is $k^{2^{O(k)}} O([\Gamma])$.

Open Problems

What will happen if more points are given? Given k sets of points $S_1, \ldots, S_k$ on a grid with $\forall i \in [n], |S_i| \geq 2$, how to find k vertex-disjoint paths that connects each points in a set with the total length of the solution to be minimized?

1 Other topics

2 Online Algorithms

- Online Robust Mean Estimation
- Dynamic algorithms for k-center on graphs
- Bin Packing under Random-Order: Breaking the barrier of $\frac{3}{2}$

Online Robust Mean Estimation

Background

上海交通大學 SHANGHAI JIAO TONG UNIVERSITY

Consider a general offline setting first. Given n samples from a distribution $\mathcal{X}$ on $\mathbb{R}^m$ with unknown mean μ^*. The algorithm should return an estimate $\hat{\mu}$ of μ^*, and make $||\hat{\mu} - \mu^*||$ as small as possible.
However, as is known to all, some samples are generated by malicious users, especially competitors, and thus a fraction of data might be misleading or meaningless.
A formal definition of this process is given below.

Definition 1.1.1 (ε-corrupted)

Given a parameter $0 \leq \varepsilon \leq \frac{1}{2}$ and a set $\mathcal{C}$ of n samples, an ε-corrupted version of $\mathcal{C}$ is generated after an adversary removes up to εn samples from $\mathcal{C}$ and replace them by arbitrary points.

Settings

Given $M, T \in \mathbb{Z}^+$, *s.t.* M is an interger multiple of T, $0 \leq \varepsilon < \frac{1}{2}$, let $\mathcal{X} = \{x^{(1)}, \ldots, x^{(n)}\}$ be an ε-corrupted version of a clean set of *i.i.d.* samples from a distribution X on $\mathbb{R}^M$ with unknown mean μ^*. The M coordinates of each data point are divided into T batches, each of size $d \triangleq \frac{M}{T}$, *i.e.* $x^{(i)}$ is the concatenation of $x_1^{(i)}, \ldots, x_T^{(i)}$, where $x_t^{(i)} \in \mathbb{R}^d$, $t \in [T]$. The interaction with the learner proceeds in T rounds as follows:

1. In the t-th round, the t-th batch of coordinates $x_t^{(1)}, \ldots, x_t^{(n)} \in \mathbb{R}^d$ are revealed.
2. After the t-th round, the algorithm is required to output $\hat{\mu}_t \in \mathbb{R}^d$ at an estimate of μ_t^*- the t-th batch of coordinates of μ^*.

Settings

At the end of this process, we say that the algorithm estimates the mean of $\mathcal{X}$ under ε-corruption in the T-round online setting with error $\varepsilon' > 0$, failure probability $\tau \in (0, 1)$ and sample complexity n if

$$\Pr[||\hat{\mu} - \mu^*||_2 - \varepsilon' \leq 0] = \Pr[\sqrt{\sum_{t=1}^{T} ||\hat{\mu}_t - \mu_t^*||_2^2} - \varepsilon' \leq 0] \geq 1 - \tau.$$

Idea

This problem can be understood as calculating a weight for each sample. Suppose an algorithm generates a set of weights $w_t^{(i)}$ for each sample $i \in [n]$ at the end of round t. Then initialize the weights of the filtering algorithm in the $(t+1)$-th round to $w_t^{(i)}$. Unfortunately, the idea of simply applying the original weights and the current new input to update weights is theoretically not feasible. Consider the case where the unknown distribution X is the product of T isotropic Gaussians $X_1, \ldots, X_T$. Then, the adversary can contaminate the sample set $\mathcal{C}$ to make them look like *i.i.d.* samples from another isotropic Gaussian distribution X'_t at each round $t \in [T]$, such that $||\mathrm{E}[X'_t] - \mathrm{E}[X_t]||_2 = c \cdot \varepsilon$ for some constant c. Then, the filtering algorithm should not downweight any sample, since the coordinates revealed in each round of contaminated samples are statistically indistinguishable from the *i.i.d.* samples from X'_t. Therefore, the algorithm's error still increases with $\sqrt{T}$, but the weights remain constant throughout.

Algorithm

SHANGHAI JIAO TONG UNIVERSITY

Algorithm 1 Online Filter

Input: The number of samples n, Byzantine fraction ϵ, round number T, sample coordinates $x_t^{(1:n)}$ revealed at the t^{th} round for $t = 1, 2, \cdots, T$, filter threshold λ, and initialized weight $w_0^{(i)} = 1/n$, for $i = 1, 2, \cdots, n$.
for $t = 1, 2, \cdots, T$ **do**
  Initialize $\boldsymbol{w}_t \leftarrow \boldsymbol{w}_{t-1}$ and update $\bar{\boldsymbol{x}}_t^{(i)} = (x_1^{(i)}, ..., x_t^{(i)})$.
  Compute $\Sigma \leftarrow \mathrm{WCov}(\boldsymbol{w}_t, \bar{\boldsymbol{x}}_t^{(1:n)})$.
  while $\|\Sigma\|_2 > 1 + \lambda$ **do**
    Compute the top eigenvector v of Σ.
    Compute empirical weighted mean $\mu(\boldsymbol{w}_t) \leftarrow \sum_{i=1}^n w_t^{(i)} \bar{\boldsymbol{x}}_t^{(i)}$.
    for $i = 1, 2, \cdots, n$ **do**
      Compute $\rho^{(i)} \leftarrow \langle v, \boldsymbol{x}^{(i)} - \mu(\boldsymbol{w}_t)\rangle^2$.
    end for
    Sort $\rho^{(1:n)}$ into a decreasing order denoted as $\rho^{\pi(1)} \geq \rho^{\pi(2)} \geq \cdots \geq \rho^{\pi(n)}$, and let β be the smallest number such that $\sum_{i=1}^{\beta} \rho^{\pi(i)} > 2\epsilon$.
    Apply WFilter to update weights for $\{\pi(1), \cdots, \pi(\beta)\}$ as $\{w_t^{\pi(1)}, \cdots, w_t^{\pi(\beta)}\} \leftarrow \mathrm{WFilter}(\rho^{\pi(1:\beta)}, w_t^{\pi(1:\beta)})$; while the remaining weights keep the same, i.e., $w_t^{\pi(i)} = w_t^{\pi(i)}$ for $i > \beta$.
    Update $\Sigma \leftarrow \mathrm{WCov}(\boldsymbol{w}_t, \bar{\boldsymbol{x}}_t^{(1:n)})$
  end while
  Output: $\mu_t = \sum_{i=1}^n w_t^{(i)} x_t^{(i)}$.
end for

Subroutine 1: Weighted Filter (WFilter)

Input: scores $\rho^{\pi(1:\beta)}$, weights $w^{\pi(1:\beta)}$.
for $i = 1, 2, ..., \beta$ **do**
  $w^{\pi(i)} \leftarrow (1 - \frac{\rho^{\pi(i)}}{\max_j \rho^{\pi(j)}}) w^{\pi(i)}$.
end for
Output: $w^{\pi(1:\beta)}$.

Subroutine2: Weighted Covariance (WCov)

Input: weight $\boldsymbol{w} = w^{(1:n)}$ and samples $\bar{x}^{(1:n)}$.
Compute $\mu(\boldsymbol{w}) \leftarrow \sum_{i=1}^n \frac{w^{(i)}}{\|\boldsymbol{w}\|_1} \bar{x}^{(i)}$.
Compute weighted covariance estimation $\Sigma \leftarrow \sum_{i=1}^n \frac{w^{(i)}}{\|\boldsymbol{w}\|_1} (\bar{x}^{(i)} - \mu(\boldsymbol{w}))(\bar{x}^{(i)} - \mu(\boldsymbol{w}))^T$.
Output: Σ.

Result

Definition 1.1.2 ((ε, δ)-stability)

For $\varepsilon \in (0, \frac{1}{2})$ and $\delta \geq \varepsilon$, a finite set $\mathcal{S} \subset \mathbb{R}^d$ is (ε, δ)-stable with respect to a vector $\mu \in \mathbb{R}^d$ if for every unit vector $v \in \mathbb{R}^d$ and every subset $\mathcal{S}' \subseteq \mathcal{S}$, where $|\mathcal{S}'| \geq (1-\varepsilon)|\mathcal{S}|$, the following conditions are satisfied:

1. $\left| \frac{1}{|S'|} \sum_{x \in S'} v^T \cdot (x - \mu) \right| \leq \delta$.
2. $\left| \frac{1}{|S'|} \sum_{x \in S'} \left(v^T \cdot (x - \mu)\right)^2 - 1 \right| \leq \frac{\delta^2}{\varepsilon}$

Suppose that $\mathcal{C}$ is (ε, δ)-stable with respect to μ^* and $\mathcal{X}$ is an ε-corrupted version of $\mathcal{C}$. Then, for ε at most a sufficiently small positive constant, there exists some constant κ such that their algorithm, when $\lambda = \frac{\kappa\delta^2}{\varepsilon}$, outputs a sequence of estimates satisfying $||\hat{\mu} - \mu^*||_2 = O(\delta \log T)$.

Result

If the distribution is more structured, there is an algorithm that achieves the error with no dependence on T whatsoever.

Dynamic algorithms for k-center on graphs

Settings

Given a metric space with n points and a positive integer $k \leq n$, the goal of the k-center problem is to select k points, referred to as centers, such that the maximum distance of any point in the metric space to its closest center is minimized.
The formal distribution is given below:
Given a weighted undirected graph $G = (V, E, w)$ and an integer $k \geq 1$, the goal is to output a subset of vertices $S \subseteq V$ of size at most k, such that the value $\max_{v \in V} d_G(v, S)$ is minimized.
In the online setting, edges can be inserted into G or deleted from G. And the goal is to minimize the time complexity in each update and achieve better performance.

Results

Given a weighted undirected graph $G = (V, E, w)$ subject to edge updates, an integer parameter $k \geq 1$, and a positive constant parameter $\varepsilon \leq \frac{1}{2}$, there are two fully dynamic algorithms for the k-center problem on graphs, that maintain a $(2 + \varepsilon)$-approximation with the following guarantees (based on the current value of the matrix multiplication exponent):

1. Deterministic algorithm with $O(kn^{1.529}\varepsilon^{-2})$ worst-case update time, if G has uniform weights;
2. Randomized algorithm, against an adaptive adversary, with $O(kn^{1.823}\varepsilon^{-2})$ worst-case update time, if G has general weights.

Both algorithms have preprocessing time $O(n^{2.373}\varepsilon^{-2}\log\varepsilon^{-1})$.

Results

Two special online setting are also concerned.

1. For the incremental setting, in which only edge insertions are allowed, given a weighted undirected graph $G = (V, E, w)$ subject to edge insertions, an integer parameter $k \geq 1$, and a positive constant parameter $\varepsilon < 1$, there is a randomized incremental $(4 + \varepsilon)$-approximation algorithm for the k-center problem on graphs. The algorithm is correct *w.h.p.* and has $kn^{o(1)}$ amortized update time over a sequence of $\Theta(m)$ updates.
2. For the decremental setting, in which only edge deletions are allowed, given a weighted undirected graph $G = (V, E, w)$ subject to edge deletions, an integer parameter $k \geq 1$, and a positive constant parameter $\varepsilon < 1$, there is a deterministic decremental $(2 + \varepsilon)$-approximation algorithm for the k-center problem on graphs, with $kn^{o(1)}$ amortized update time over a sequence of $\Theta(m)$ updates.

Bin Packing under Random-Order: Breaking the barrier of $\frac{3}{2}$

Settings

Given a list $I := (x_1, \ldots, x_n)$, where $\forall i \in [n], x_i \in (0,1]$, the goal is to partition them into minimum number of unit size bins. In the online setting, item sizes are revealed one by one: in round i, the item x_i arrives and needs to be irrevocably assigned to a bin before the next items are revealed.

For a deterministic algorithm $\mathcal{A}$, the performance can be measured as the competitive ratio $R^{\infty}_{\mathcal{A}} = \limsup_{m\to\infty}\left(\sup_{I:OPT(I)=m}\left(\frac{\mathcal{A}(I)}{OPT(I)}\right)\right)$, where $\mathcal{A}(I)$ denotes the number of bins used by $\mathcal{A}$ to pack an input sequence I.

Settings

For the online setting with the random-order model, the input set of items is chosen from adversary and the arrival order of the items is decided according to a permutation chosen uniformly at random from $\mathcal{S}_n$, the set of permutations of n elements. This reshuffling of the input items often weakens the adversary and provides better performance guarantees. In this model, the performance of an online algorithm $\mathcal{A}$ can be measured using the following quantity, called random-order ratio (RR):

$$RR_{\mathcal{A}}^{\infty} = \lim_{m\to\infty} \sup \left(\sup_{I:OPT(I)=m} \left(\frac{\mathbf{E}_\sigma[\mathcal{A}(I_\sigma)]}{OPT(I)}\right) \right)$$

History

Best-Fit algorithm is one of the most widely-used algorithms for bin packing. The idea is natural, simple and behaves well in practice. Best-Fit packs each item into the fullest bin where it fits, and open a new bin only if the item fits into none of the present opening bins.

It's proved that the RR of Best-Fit is upper-bounded by $\frac{3}{2}$ and lower-bounded by 1.08. And in that paper, the author conjectured that the true ratio should lie somewhere close to 1.15.

Results

1. Shows that Best-Fit achieves a RR of at most $1.5 - \varepsilon$ for a small constant $\varepsilon > 0$.
2. Improve the lower bound of random order ratio to 1.144.

Technique

Consider the last time t_σ of putting an item of size at most $\frac{1}{3}$ in a bin of load at most $\frac{1}{2}$. Previous result shows that all bins except at most one that opened by Best-Fit to pack the first t_σ items are filled up to the level of at least $\frac{2}{3}$. And to pack items arriving after t_σ, Best-Fit is within a $\frac{3}{2}$ factor of OPT. Combining these observations, an upper bound of $\frac{3}{2}$ was achieved.

To break the barrier, they did further analysis that either the factor of the period before t_σ can be improved to $\frac{3}{2} - 2\varepsilon$, or the factor after t_σ can be improved to $\frac{3}{2} - 2\varepsilon$. They did a case analysis based on t_σ and show that either a large fraction of the bin packed by Best-Fit is rather full (the load is at least $\frac{3}{4}$) or Best-Fit performs relatively well compared to OPT.

谢 谢