MAX-MIN NOTES

Improved Approximation and scalability for Fair Max-Min Diversification

Motivation:

- Consider a query over a maps service for finding restaurants
  around Manhattan at NYC. Then the goal is to present the user with a diversified set of restaurant locations while representing different cuisines in the sample.
- The aim is to construct a diverse set of points where each group is sufficiently represented.

Definitions Section

- Max-Min Diversification
  - $div(\mathcal{S}) = \min_{u,v \in S, u \neq v} d(u,v)$
  - objective function: $\max_{\mathcal{S} \subseteq \mathcal{X}} \ div(\mathcal{S})$
  - OPT: let $\mathcal{S}^* = \bigcup_{i=1}^{m} \mathcal{S}_i^*$ be the set of points that obtains the optimal diversity score denoted by $div(\mathcal{S}) = l^*$
  - α approximation & β fairness:
    - if $div(S) \geq l^*/\alpha$, which is $l^* \leq \alpha \ div(S)$
    - if $|S \cap \mathcal{X}_i| \geq \beta k_i$ for all $i \in [m]$. When $\beta = 1$, we say the subset achieves perfect fairness
- Coresets(useful for distributed computing):
  - **coreset for fair max-min**: A set $\mathcal{T} \subseteq \mathcal{X}$ is an α-coreset if there exists a subset $\mathcal{T}' \subseteq \mathcal{T}$ with ($|\mathcal{T}' \cap \mathcal{X}_i| = k_i, \forall i \in [m]$) and $l^* \leq \alpha \ div(\mathcal{T}')$
  - **composable coresets:**
    - **Composable cooreset for fair max-min**: A function $c(\mathcal{X})$, which maps a set of element to a subset of these elements, computes an α-composable coreset for some $\alpha \geq 1$, if for any partitioning of $\mathcal{X} = \bigcup_j \mathcal{Y}_j$ and $\mathcal{T} = \bigcup_j c(\mathcal{Y}_j)$, there exists a set $\mathcal{T}' \subseteq \mathcal{T}$ with $|\mathcal{T}' \cap \mathcal{X}_i| = k_i, \ \forall i \in [m]$ s.t. $div(\mathcal{T}') \geq l^*/\alpha$
    - (from wikipedia) if we have dataset D_1, D_2, D_3, and coresets C_1, C_2, C_3 respectively, then the combine of the coresets $C_1, C_2.C_3$ is the coreset of D_1, D_2, D_3

- Low Doubling Dimension Spaces: Let $(\mathcal{X}, d)$ be a metric space. The doubling dimension of $\mathcal{X}$ is the smallest integer λ such that any ball $B(p, r)$ of radius r around a point $p \in \mathcal{X}$ can be covered using at most $(r/r')^\lambda$ balls of radius r'. The Euclidean metric on $\mathbb{R}^D$ has doubling dimension $O(D)$.

Algorithms Section

- **3.1 Expected Fairness & Guarentee 2 approximation**
  - Basic idea: randomized algorithm + (ILP relaxation → LP), with some expectation calculation, basically we are selecting a candidate in a specfic probabilty and also make sure that they are far away (specifically, by $\gamma/2$).
  - Worth to mention: Because there are at most (n choose 2) option for the solution γ, so this algorithm would run at most $O(n^2)$ times for a valid 2-approx solution.
  - First assume and guess a γ as the optimal diversity value
  - and here is the LP construction

    $$\sum_{pj \in \mathcal{X}_i} x_j \geq k_i \quad \forall i \in [m].$$
    $$\sum_{p_\ell \in \mathrm{B}(p, \gamma/2)} x_\ell \leq 1 \quad \forall p \in \mathcal{X}.$$
    $$x_j \geq 0 \quad \forall j \in [n].$$

    where x_j is a number in $[0, 1]$, represents for if node j “in the coreset” or not. used in first constraint a.k.a. fairness constraint

    Question: Why is the first summation greater or equal to k rather than exactly equal to k? **Answer**: In this scenario, we want to choose less point. Otherwise we will have a worse max-min. So it doesn’t matter if we’ve chosen more points. I think we can just drop the point that violates the constraint, aka have more points than we need.
- Algorithm (summarized pseudo code)
  - Pick a γ which as large as possible but satisfied $\gamma \leq l^*$
  - Solve the LP
  - Rounding Process
    - we have indicator x_j^* for every points. And let $n' = |\{j : x_j > 0\}|$ (which means that we don’t care point j with $x_j \leq 0$
    - generate a sequence with length n', sampling without replacement, follow the probabilty that $P[\sigma(t) = j] = \frac{x_j^*}{\sum_{l \in R_t} x_l^*}$ (the given point appears in the solution set)

- R_t is $[n'] \setminus \{\sigma(1), ..., \sigma(t-1)\}$
- n' is the number of the element with $x_j^* > 0$
- After generate the sequence, we have the solution set by this operation: We would like to include the point p_j in it if and only if $\sigma(j) \leq \sigma(l)$ for all p_l in $B(p_j, \gamma/2)$
- $$\begin{aligned}\Pr\left[p_j \in \mathcal{S}\right] = \sum_{t=1}^{n'} \Pr\left[\sigma(t) = j \mid A_t\right] \Pr\left[A_t\right] &= \sum_{t=1}^{n'} \frac{x_j^*}{\sum_{p_\ell \in \mathbf{B}(p_j, \gamma/2)} x_\ell^*} \Pr\left[A_t\right] \\ &= \frac{x_j^*}{\sum_{p_\ell \in \mathbf{B}(p_j, \gamma/2)} x_\ell^*} \sum_{t=1}^{n'} \Pr\left[A_t\right] \\ &= \frac{x_j^*}{\sum_{p_\ell \in \mathbf{B}(p_j, \gamma/2)} x_\ell^*} \geq x_j^*\end{aligned}$$
- A_t is the event $d(p_{\sigma(t)}, p_j) < \gamma/2$ and $d(p_{\sigma(t')}, p_j) \geq \gamma/2$
- the last inequality holds since $\sum_{t=1}^{n'} \Pr[A_t] = 1$, and with the second constraint.

- **3.2 Guarentee 6-Approx & Guarentee $(1-\epsilon)$ fairness**
  - Basic idea: Based on 3.1, give one more non-linear constriant to guarantee the fairness constraint. (The other change is that change the radius of the ball from $\gamma/2$ to $\gamma/6$
  - "LP" Construction

$$\begin{gathered}\sum_{p_j \in \mathcal{X}_i} y_j \geq k_i \quad \forall i \in [m]. \\ \sum_{p_\ell \in \mathbf{B}(p, \gamma/6)} y_\ell \leq 1 \quad \forall p \in \mathcal{X}. \\ y_j \geq 0 \quad \forall j \in [n]. \\ (0 < y_i \text{ and } 0 < y_j) \Rightarrow d\left(p_i, p_j\right) \geq \tfrac{\gamma}{3} \quad \forall p_i, p_j \in \mathcal{X}_\ell, \forall \ell \in [m]\end{gathered}$$

  - HOW construct solution set $\{y_j^*\}_{j \in [n]}$ from $\{x_j^*\}_{j \in [n]}$:

    (a) For each $p_j \in \mathcal{X}$ with $x_j^* > 0$ satisfying $p_j \in \mathcal{X}_i$ *and* y_j^* value not yet set, we set:

$$y_j^* \leftarrow \left(\sum_{p_\ell \in \mathbf{B}\left(p_j, \frac{\gamma}{3}\right) \cap \mathcal{X}_i} x_\ell^*\right) \text{ and } y_\ell^* \leftarrow 0 \text{ for all } p_\ell \in \mathbf{B}\left(p_j, \tfrac{\gamma}{3}\right) \cap \left(\mathcal{X}_i \backslash \{p_j\}\right)$$

    (b) Finally, for all $p_j \in \mathcal{X}$ with $x_j^* = 0$, we set $y_j^* \leftarrow 0$.
    Informally, we are just moving weight to p_j from points of the same group (as) that are at a distance strictly less than $\gamma/3$ from .
  - Proof:
    - 1. constructed set satisfied the "LP" 4 constraints

(TODO: Need to rephrase here)

      - 2. fairness constraint satisfied (proved by constraint 4)

        (TODO: Need to rephrase here)

  - Rounding: Create the permutation σ as previous. Add p_j to the output $\mathcal{S}$ if $\sigma(j) \leq \sigma(\ell)$ for all p_ℓ such that $d(p_\ell, p_j) < \gamma/6$.
  - Theorem 7.

    Asssume $k_i \geq 3\epsilon^{-2}\log(2m)$ for all $i \in [m]$. There is a poly $\left(n, k, \delta^{-1}\right)$ time algorithm that returns a subset of points with diversity $\ell^*/6$ and includes $(1-\epsilon)k_i$ points in each group $i \in [\mathrm{m}]$ with probability at least $1-\delta$.

    Proof: (TODO: Need to rephrase here)

- **3.3 $(m+1)$-Approx with Perfect Fairness (where $\beta = 1$)**
  - Basic idea: greedy, max-flow
  - Algorithm:

Algorithm 1 FAIR-GREEDY-FLOW.

Input: $\mathcal{X} = \bigcup_{i=1}^{m} \mathcal{X}_i$: Universe of available elements.
$k_1, \ldots, k_m \in \mathbb{Z}^+$.
$\gamma \in \mathbb{R}^+$: A guess of the optimum fair diversity.
Output: k_i points in $\mathcal{X}_i$ for $i \in [m]$.

$\mathcal{R} \leftarrow \mathcal{X}$ denote the set of remaining elements.
$\mathcal{C} \leftarrow \emptyset$ denote a collection of subsets of points (called clusters).
while $|\mathcal{R}| > 0$ **(and)** $|\mathcal{C}| \leq km$ **do**
  $D \leftarrow \emptyset$ denote the current cluster, and $D_{\text{col}} \leftarrow \emptyset$ denote the groups of points in cluster D.
  while an element $p \in \mathcal{R} \cap \mathcal{X}_i$ for some $i \in \{1, 2, \cdots, m\} \setminus D_{\text{col}}$. exists **do**
    if $|D| = 0$ (or) $d(p, x) < \frac{\gamma}{m+1}$ for some $x \in D$ **then**
      $D \leftarrow D \cup \{p\}$ and $D_{\text{col}} \leftarrow D_{\text{col}} \cup \{i\}$.
    end if
  end while
  $\mathcal{R} \leftarrow \mathcal{R} \setminus \bigcup_{p \in D} \mathbf{B}(p, \frac{\gamma}{m+1})$.
  $\mathcal{C} \leftarrow \mathcal{C} \cup \{D\}$.
  $\mathcal{R} \leftarrow \mathcal{R} \setminus \mathcal{X}_i \ \forall i \in [m]$ if $|\{D \mid D \in \mathcal{C} \text{ and } D \cap \mathcal{X}_i \neq \emptyset\}| \geq k$.
end while
▷Construct flow graph :
Let $\mathcal{C} = \{D_1, D_2, \cdots D_t\}$.
Construct directed graph $G = (V, E)$ where

$$\begin{aligned} V &= \{a, u_1, \ldots, u_m, v_1, \ldots, v_t, b\} \\ E &= \{(a, u_i) \text{ with capacity } k_i : i \in [m]\} \\ &\cup \{(v_j, b) \text{ with capacity } 1 : j \in [t]\} \\ &\cup \{(u_i, v_j) \text{ with capacity } 1 : |\mathcal{X}_i \cap D_j| \geq 1\} \end{aligned}$$

Set $\mathcal{S} \leftarrow \emptyset$. Compute maximum a-b flow in G using Ford-Fulkerson algorithm [26].
if flow size $< k = \sum_i k_i$ **then return** $\emptyset$ ▷Abort
else ▷max flow is k
  $\forall (u_i, v_j)$ with flow equal to 1, add the point in D_j with group i to $\mathcal{S}$.
end if
return $\mathcal{S}$.

- Tight example:

  (TODO: Need to rephrase here)

- **3.4 Hardness for the approx: via a reduction from GAP-CLIQUE$_\rho$, which is the NP-Hard problem, leads to the open question**
- Euclidean Metrics
  - When $p_i \in \mathbb{R}^D$, $D = 1$, Dynamic Programming
  - More generally, $D = O(1)$ we present a bi-criteria approximation that uses an **extension of the dynamic programming** approach and **properties of low dimensional Euclidean spaces**.

Related work Section

- When the number of groups $m = 1$, it is called facility location, information retrieval, web search and recommandation systems.

- Diversity in Big Data: A Review
  - Give the use cases of the diversity representing: Hiring, Matchmaking, Search and content recommendation
  - MAx-SUM or MAX-MIN(aka p-dispersion, if the p = 1 it could be solved opt in poly)
  - so we have two way to classify the “coverage”. 1-for each group, we need to choose k_i point for representation(cover all the group). 2-for all point $i \in \mathcal{I}$, we have $j \in \mathcal{S}$ s.t. $d(i, j) \leq r$ (cover all the point)
