Balanced Popularity in Multi-Product Billboard Advertisement

Dildar Ali, Suman Banerjee, and Yamuna Prasad

Department of Computer Science and Engineering, Indian Institute of Technology Jammu, Jammu & Kashmir-181221, India.
{2021rcs2009,suman.banerjee,yamuna.prasad}@iitjammu.ac.in

Abstract. The billboard advertisement has emerged as an effective out-of-home advertisement technique where the objective is to choose a limited number of slots to play some advertisement content (e.g., animation, video, etc.) with the hope that the content will be visible to a large number of travelers, and this will be helpful to earn more revenue. In this paper, we study a variant of the influential slot selection problem where the advertiser wants to promote multiple products. Formally, we call this problem the Multi-Product Influence Maximization Problem for the Balanced Popularity Problem. The input to our problem is a trajectory and a billboard database, as well as a budget for each product. The goal here is to choose a subset of slots for each product such that the aggregated influence of all the products gets maximized subject to the following two constraints: total selection cost for each product is less than or equal to the allocated budget for that product, and the difference between the influence for any two products is less than or equal to a given threshold. We show that the problem is NP-hard to solve optimally. We formulate this problem as a linear programming problem and use linear programming relaxation with randomized rounding. Further, we propose a greedy-based heuristic with balance correction to solve this problem. We conduct a number of experiments with real-world trajectory and billboard datasets, and the results are reported. From the reported results, we observe that the proposed solution approaches lead to more influence compared to many baseline methods.

Keywords: Billboard Advertisement, Influence, Slot, Product, Budget

1 Introduction

Recently, *Billboard Advertisement* has emerged as an effective out-of-home advertising technique due to the guaranteed return on investment[1]. In a billboard advertisement, a number of digital billboards are placed across popular places in a city, and the advertisement contents (e.g., animation, video, etc.) are played with the hope that the content will be observed by many people. This is a popular out-of-home advertisement technique that is helpful to reach a large audience

[1] https://www.lamar.com/howtoadvertise/Research/

with a low budget to create influence and awareness of the brand. Eventually, a little investment in advertising may lead to a large revenue with high probability.

Background. To our knowledge, Zhang et al. [14,15] are the first to introduce the influence maximization problem in billboard advertising. This problem has been studied in the literature, and a number of solution methodologies have been proposed, such as the graph-based pruned-submodularity approach [1], the spatial clustering approach [2], the branch-and-bound approach [3,16], and many more. Wang et al. [13] introduce the problem of finding the k best advertising units in billboard advertisements. There also exists literature in the context of advertising on billboards [6,7] and social networks [11] that select tags and slots jointly. Deviating from the influence maximization, there exists literature [4,8,17] that considers regret minimization in billboard advertising. In practice, an e-commerce house has multiple products to advertise. However, to our knowledge, there is no literature that considers slot selection for multiple products. Hence, in this paper, we address the Multi-Product Influence Maximization Problem for the Balanced Popularity Problem.

Motivation. Consider the scenario where a commercial house wants to promote more than one product through billboard advertising. Different products will have different features, and hence it may not be meaningful to promote all the products to the same set of people. To make the campaign more effective, it is important to select slots that cover the people who are more likely to like the product. In this paper, we study the problem of selecting slots for multiple products where the goal is to maximize the aggregated influence for all products. However, it is important to observe that it may not be beneficial for the commercial house if there is a significant difference in influence between any two products. Because if for any product its influence is very low compared to another product, then the expected sales of the product will be low, and the initial investment of the commercial house for launching the product will go in vain. As a remedy, the commercial house may wish to choose slots in such a way as to lead to a balanced influence for all the products.

Our Contribution. To the best of our knowledge, this problem has not been studied previously, and we make the following contributions.

- We study the noble problem called the Multi-Product Influence Maximization Problem for the Balanced Popularity problem for which there is no literature.
- We formulate this problem as a linear programming problem and use linear programming relaxation with randomized rounding. Further, we propose a greedy-based heuristic with balance correction to solve this problem.
- We conducted experiments with real-world trajectory and billboard datasets to show the effectiveness and efficiency of the proposed solution approach.

Organization of the Paper The rest of the paper has been organized as follows. Section 2 describes the background information and defines the problem formally. The proposed solution approaches have been described in Section 3. Section 4 describes the experimental evaluation of the proposed solution approaches. Section 5 describes the concluding remarks of our study.

2 Background and Problem Definition

Trajectory and Billboard Database. A trajectory database contains the location information of a set of people in a city across different time stamps. In our problem, a trajectory database $\mathcal{D}$ contains tuples of the form $(u_i, \texttt{loc}, [t_a, t_b], \mathcal{P}(u_i))$ and this signifies that the person u_i is at the location `loc` for the duration t_a to t_b. Here, $\mathcal{P}(u_i)$ denotes the set of products in which user u_i will be interested. For any person u_i, $loc_{[t_x,t_y]}(u_i)$ denotes the locations of u_i during the time interval $[t_x, t_y]$. Let $\mathcal{U}$ denote the set of people whose movement data is contained in $\mathcal{D}$. Let, $T_1 = \min\limits_{\tau \in \mathcal{D}} t_a$ and $T_2 = \max\limits_{t \in \mathcal{D}} t_b$ and we say that the trajectory database $\mathcal{D}$ contains the movement data from time stamp T_1 to T_2. A billboard database $\mathbb{B}$ contains the information about billboard slots. Typically, this contains the tuples of the following form $(b_{id}, s_{id}, \texttt{loc}, \texttt{duration})$ where b_{id} and s_{id} denote the billboard id and slot id. The `loc` and `duration` signify the billboard's location and the slot's duration.

Billboard Advertisement. Let a set of m billboards $\mathcal{B} = \{b_1, b_2, \ldots, b_m\}$ be placed in different locations of a city. We assume that all the billboards can be leased for a multiple of a fixed duration Δ, which is called a 'slot' and has been stated in Definition 1.

Definition 1 (Billboard Slot). *A billboard slot is defined by a tuple consisting of two entities, the billboard ID and the duration, that is,* $(b_{id}, [t, t + \Delta])$.

The set of all billboard slots is denoted by $\mathcal{BS}$, i.e., $\mathcal{BS} = \{(b_i, [t, t + \Delta]) : i \in \{1, 2, \ldots, m\}$ and $t \in \{T_1, T_1 + \Delta, T_1 + 2\Delta, \ldots, T_2 - \Delta\}\}$. It can be observed that $|\mathcal{BS}| = m \cdot \frac{T}{\Delta}$ where $T = T_2 - T_1$. For simplicity, we assume that T is perfectly divisible by Δ. For any slot $s_j \in \mathcal{BS}$, b_{s_j} denotes the corresponding billboard and $[t^s_{s_j}, t^f_{s_j}]$ denotes the time interval for the slot s_j where $t^f_{s_j} = t^s_{s_j} + \Delta$. Now, we state the notion of influence probability of a billboard slot in Definition 2.

Definition 2 (Influence Probability of a Billboard slot). *Given a slot* $s_j \in \mathcal{BS}$ *and a person* $u_i \in \mathcal{U}$*, the influence probability of* s_j *on* u_i *is denoted by* $Pr(s_j, u_i)$ *and can be computed using the following conditional equation.*

$$Pr(s_j, u_i) = \begin{cases} \frac{size(s_j)}{\max\limits_{s_k \in \mathcal{BS}} size(b_{s_k})}, & \text{if } loc_{[t^s_{s_j}, t^f_{s_j}]}(u_i) = loc(s_j) \\ 0, & \text{otherwise} \end{cases}$$

Now, we define the influence of a subset of billboard slots in 3.

Definition 3 (Influence of Billboard Slots). *Given a trajectory database $\mathcal{D}$, and a subset of billboard slots $\mathcal{S} \subseteq \mathcal{BS}$, the influence of $\mathcal{S}$ can be defined as the expected number of trajectories that are influenced, which can be computed using Equation 1.*

$$\mathcal{I}^{'}(\mathcal{S}) = \sum_{u_i \in \mathcal{U}} [1 - \prod_{s_j \in \mathcal{S}} (1 - Pr(s_j, u_i))] \tag{1}$$

Here, $Pr(b_j, u_i)$ denotes the influence probability of the billboard slot b_j on the people u_i, and $\mathcal{I}^{'}(\mathcal{S})$ denotes the influence of the billboard slots of $\mathcal{S}$. $\mathcal{I}^{'}()$ is the influence function which maps each possible subset of billboard slots to its corresponding influence value, i.e., $\mathcal{I}^{'} : 2^{\mathcal{BS}} \longrightarrow \mathbb{R}_0^+$ where $\mathcal{I}^{'}(\emptyset) = 0$.

Now, it is important to observe that every product may not be relevant to every trajectory user. The effective advertisement requires targeting the right users. Let $\mathcal{P} = \{p_1, p_2, \ldots, p_\ell\}$ be the set of products. From the trajectory database, each user's relevant products are known. For any product $j \in [p]$, we define Product Specific Influence for a billboard slot in Definition 4.

Definition 4 (Product Specific Influence). *Given a specific product and a subset of billboard slots $\mathcal{S} \subseteq \mathcal{BS}$, the product-specific influence of $\mathcal{S}$ for the product j is denoted by $\mathcal{I}_j(\mathcal{S})$ and can be computed using Equation 2.*

$$\mathcal{I}_j(\mathcal{S}) = \sum_{u_i \in \mathcal{U}_k} [1 - \prod_{s_j \in \mathcal{S}} (1 - Pr(s_j, u_i))] \tag{2}$$

Here, $\mathcal{U}_k$ denotes the set of users relevant to the k-th product and $\mathcal{I}$ is the product specific influence function, i.e., $\mathcal{I} : 2^{\mathcal{BS}} \times \mathcal{P} \longrightarrow \mathbb{R}_0^+$.

Problem Definition. Consider that a commercial company wants to promote ℓ many products through a billboard advertisement. The input to our problem is a trajectory and billboard database denoted by $\mathcal{D}$ and $\mathbb{B}$, respectively, ℓ positive integers $k_1, k_2, \ldots, k_\ell$ and a threshold value θ. The goal of this problem is to find an allocation $(\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_\ell)$ where $\mathcal{S}_i$ denotes the allocated slots for the i-th product for all $i \in [\ell]$. Here, the objective is to maximize the sum of the influences of all products, i.e., to maximize $\sum_{i \in [\ell]} I_i(\mathcal{S}_i)$. The constraints involved in this problem are as follows:

- **Budget Constraint**: This constraint ensures that for any product the number of allocated slots should be less than the corresponding budget, i.e., for all $i \in [\ell]$, $|\mathcal{S}_i| \leq k_i$.
- **Non-Intersecting Slot Constraint**: This constraint ensures that for any $i, j \in [\ell]$ and $i \neq j$, $\mathcal{S}_i \cap \mathcal{S}_j = \emptyset$.
- **Influence Difference Constraint**: This constraint ensures that the influence between any two products will be less than equal to a given threshold, i.e., for all $i, j \in [\ell]$, and $i \neq j$, $|I_i(\mathcal{S}_i) - I_j(\mathcal{S}_j)| \leq \theta$ where θ is a given threshold value.

From the computational point of view, this problem can be posed as follows:

> Multi-Product Influence Maximization Problem (MPIM)
> **Input:** A trajectory ($\mathcal{D}$) and Billboard ($\mathcal{B}$) Database, A set of billboard slots ($\mathcal{BS}$), a set of products ($\mathcal{P}$).
> **Problem:** Find a subset of slots for each product such that the aggregated influence of all the products gets maximized.

By a reduction from the Set Cover Problem [9], we can show that theMulti-Product Influence Maximization Problem is NP-hard. This result has been presented in Theorem 1. Due to the space limitation, we are not able to give the whole reduction.

Theorem 1. *The Multi-Product Influence Maximization Problem is NP-hard and hard to approximate in a constant factor.*

3 Proposed Solution Approaches

3.1 Linear Programming (LP) Relaxation

Let $\mathcal{BS}$ and $\mathcal{P}$ be the set of billboard slots and the set of products, $\mathcal{U}_i \subseteq \mathcal{U}$ be the relevant users, and k_i be the budget for the product i. $p_{s,u} \in [0,1]$ be the influence probability of the slot s on user u.

Decision variables. Let $x_{s,i} \in \{0,1\}$, where the slot s is assigned to product i and $y_{u,i} \in [0,1]$ be the auxiliary variable approximating the influence of product i on user u. The integer linear programming (ILP) formulation is stated below.

Objective (linear surrogate of expected influence).

$$\max \sum_{i \in \mathcal{P}} \sum_{u \in \mathcal{U}_i} y_{u,i}. \tag{3}$$

Constraints.

$$\begin{aligned}
&\text{(Budget)} && \sum_{s \in \mathcal{BS}} x_{s,i} \le k_i && \forall\ i \in \mathcal{P},\\
&\text{(Disjointness)} && \sum_{i \in \mathcal{P}} x_{s,i} \le 1 && \forall\ s \in \mathcal{BS},\\
&\text{(Linking / coverage upper bound)} && y_{u,i} \le \sum_{s \in \mathcal{BS}} p_{s,u}\, x_{s,i} && \forall\ i \in \mathcal{P},\ \forall\ u \in \mathcal{U}_i,\\
& && 0 \le y_{u,i} \le 1 && \forall\ i \in \mathcal{P},\ \forall\ u \in \mathcal{U}_i,\\
&\text{(Balance)} && \sum_{u \in \mathcal{U}_i} y_{u,i} - \sum_{u \in \mathcal{U}_j} y_{u,j} \le \theta && \forall\ i \ne j,\\
& && \sum_{u \in \mathcal{U}_j} y_{u,j} - \sum_{u \in \mathcal{U}_i} y_{u,i} \le \theta && \forall\ i \ne j,\\
&\text{(Integrality)} && x_{s,i} \in \{0,1\} && \forall\ s \in \mathcal{BS},\ \forall\ i \in \mathcal{P}.
\end{aligned}$$

Remark. The constraint $y_{u,i} \leq \sum_s p_{s,u} x_{s,i}$ is the linear upper bound for the nonlinear probability $1 - \prod_s (1 - p_{s,u})$. We also relax $x_{s,i} \in \{0, 1\}$ to $x_{s,i} \in [0, 1]$ and solve linear programming to obtain fractional (x^*, y^*).

3.2 Randomized rounding (RR) with feasibility repair

In our ILP formulation, we solve the relaxed LP and get a fractional allocation (x^*, y^*) where $x^*_{s,i} \in [0, 1]$ is the fractional probability of assigning a slot s to the product i. As the LP solution is not integral, we use randomized rounding [10,12] followed by a repair phase to ensure feasibility with respect to budget, disjointness, and balance constraints. First, in the randomized rounding phase, each slot is independently assigned to at most one product. Specifically, we sample a product label L_s according to the fractional values $\{x^*_{s,i}\}$. With probability $x^*_{s,i}$, the slot is assigned to product i; with probability $1 - \sum_i x^*_{s,i}$, the slot remains unassigned. Next, in the budget repair phase, some products may exceed their allocated budget (i.e., $|\mathcal{S}_i| > k_i$). To fix this, we iteratively remove excess slots that contribute the least marginal influence from the products until the budget constraint is satisfied. In the balance computation phase, we calculate the estimated influence $\widehat{\mathcal{I}}_i$ for each product based on the current slot assignments. This step allows us to check whether the influence difference constraint, $|\widehat{\mathcal{I}}_i - \widehat{\mathcal{I}}_j| \leq \theta$, is valid for all pairs of products. Next, in the balance repair phase, if the balance constraint is violated, we perform a swap/shift correction in which we identify the product with the maximum influence ($p_{\max}$) and the product with the minimum influence ($p_{\min}$), then attempt to reassign a slot from $p_{\max}$ to $p_{\min}$. The chosen slot is the one that maximizes the net gain in balancing, i.e., minimizes the loss in $p_{\max}$ while maximizing the gain in $p_{\min}$. This process is repeated until either the balance constraint is satisfied or no further beneficial swaps are possible. Finally, an allocation $(\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_\ell)$ will be return.

Complexity Analysis. Now, we analyze the time and space requirement of Algorithm 1. First, solving LP relaxation takes polynomial time in the number of slots and products, i.e., $\mathcal{O}(|\mathcal{BS}| \cdot \ell)$ with standard LP solvers. Next, in the randomized rounding phase (line no. 2 to 7), it assigns each slot independently and therefore runs in $\mathcal{O}(|\mathcal{BS}| \cdot \ell)$. In the budget repair phase (line no. 9 to 12), we may at most remove all excess slots once, giving $\mathcal{O}(|\mathcal{BS}| \cdot \ell)$ time in the worst case. In line 13 to 15, computing the product-specific influences takes $\mathcal{O}(|\mathcal{BS}| \cdot \bar{d} \cdot \ell)$, where $\bar{d}$ is the number of tuples in the trajectory database. In the balance repair phase (line no. 16 to 23), performs iterative slot shifts; in the worst case, each slot may be moved once, giving another $\mathcal{O}(|\mathcal{BS}| \cdot \bar{d} \cdot \ell)$. Hence, the total running time is bounded by $\mathcal{O}(\text{LP-solve} + |\mathcal{BS}| \cdot \ell \cdot \bar{d})$.

The Algorithm 1 requires storage for the fractional LP solution (x^*, y^*), which has $\mathcal{O}(|\mathcal{BS}| \cdot \ell)$ variables. Additionally, we maintain the slot allocation sets $(\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_\ell)$, requiring $\mathcal{O}(|\mathcal{BS}|)$ space. Storing user-specific influence values requires at most $\mathcal{O}(|\mathcal{U}| \cdot \ell)$. Therefore, the total space complexity is $\mathcal{O}\big(|\mathcal{BS}| \cdot \ell + |\mathcal{U}| \cdot \ell\big)$ or equivalently $\mathcal{O}(|\mathcal{BS}| \cdot \ell)$. Hence, Theorem 2 holds.

Algorithm 1: LP-Relaxation + RR with Balance Repair

Input: LP solution (x^*, y^*), Billboard slots $\mathcal{BS}$, products $\mathcal{P} = \{p_1, p_2, \ldots, p_\ell\}$, budgets $\{k_1, \ldots, k_\ell\}$, threshold θ, probabilities $p_{s,u}$, sets $\mathcal{U}_i$
Output: Integral allocation $(\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_\ell)$

1 $\mathcal{S}_i \leftarrow \emptyset$ for all i; ; // Initialize integral sets
2 **(A) Per-slot randomized rounding**;
3 **for** *each slot* $s \in \mathcal{BS}$ **do**
4 $\quad \pi_0 \leftarrow \max\{0,\ 1 - \sum_{i \in \mathcal{P}} x^*_{s,i}\}$; // probability of leaving s unassigned
5 $\quad$ Sample a label $L_s \in \mathcal{P} \cup \{0\}$ with $\Pr[L_s = i] = x^*_{s,i}$ and $\Pr[L_s = 0] = \pi_0$;
6 $\quad$ **if** $L_s \in \mathcal{P}$ **then**
7 $\quad\quad \mathcal{S}_{L_s} \leftarrow \mathcal{S}_{L_s} \cup \{s\}$;
8 **(B) Budget repair**;
9 **for** *each* $i \in \mathcal{P}$ **do**
10 $\quad$ **while** $|\mathcal{S}_i| > k_i$ **do**
11 $\quad\quad$ Choose $s \in \mathcal{S}_i$ with minimum loss: $\text{loss}_i(s) \leftarrow \sum_{u \in \mathcal{U}_i} \min\{p_{s,u},\ 1 - \hat{y}_{u,i}\}$, where $\hat{y}_{u,i}$ is current coverage estimate;
12 $\quad\quad$ Remove s from $\mathcal{S}_i$;
13 **(C) Balance computation**;
14 **foreach** $i \in \mathcal{P}$ **do**
15 $\quad$ Estimate influence $\widehat{\mathcal{I}}_i \leftarrow \sum_{u \in \mathcal{U}_i} \min\left\{1,\ \sum_{s \in \mathcal{S}_i} p_{s,u}\right\}$;
16 **(D) Balance repair (swap/shift)**;
17 **while** $\exists\ i, j$ *with* $|\widehat{\mathcal{I}}_i - \widehat{\mathcal{I}}_j| > \theta$ **do**
18 $\quad p_{\max} \leftarrow \arg\max_i \widehat{\mathcal{I}}_i$; $\quad p_{\min} \leftarrow \arg\min_i \widehat{\mathcal{I}}_i$;
19 $\quad$ Find $s^\star \in \mathcal{S}_{p_{\max}}$ maximizing

$$\Delta(s) = \underbrace{\sum_{u \in U_{p_{\min}}} \delta_{p_{\min}}(u; s)}_{\text{gain at } p_{\min}} - \underbrace{\sum_{u \in U_{p_{\max}}} \delta_{p_{\max}}(u; s)}_{\text{loss at } p_{\max}},$$

$\quad\quad$ where $\delta_i(u; s) = \min\{p_{s,u},\ \max(0,\ 1 - \sum_{t \in \mathcal{S}_i \setminus \{s\}} p_{t,u})\}$;
20 $\quad$ **if** $\Delta(s^\star) \le 0$ **then**
21 $\quad\quad$ **break**; ; // no beneficial shift, stop
22 $\quad \mathcal{S}_{p_{\max}} \leftarrow S_{p_{\max}} \setminus \{s^\star\}$, $\mathcal{S}_{p_{\min}} \leftarrow \mathcal{S}_{p_{\min}} \cup \{s^\star\}$;
23 $\quad$ Update $\widehat{\mathcal{I}}_{p_{\max}}, \widehat{\mathcal{I}}_{p_{\min}}$;
24 **return** $(\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_\ell)$;

Theorem 2. *The time and space requirements for Algorithm 1 will be* $\mathcal{O}(LP\text{-}solve + |\mathcal{BS}| \cdot \ell \cdot \bar{d})$ *and* $\mathcal{O}(|\mathcal{BS}| \cdot \ell)$*, respectively.*

3.3 Greedy Heuristic with Balance Correction

The Algorithm 2 starts by initializing empty slot sets $\mathcal{S}_i$ for each product $i \in \mathcal{P}$. First, in line numbers 2 to 15, in the greedy allocation phase for each product, we sample a random subset of $\left\lceil \frac{|\mathcal{BS}|}{k} \cdot \log \frac{1}{\epsilon} \right\rceil$ many candidate slots from $\mathcal{BS}$. The candidate slots assign the slot to the products with the highest influence gain. This process continues until no product has remaining budget. Next in line 16 to 30, first the influence values $\mathcal{I}_i(\mathcal{S}_i)$ are calculated for all products. If the influence difference between any two products exceeds the threshold θ, p_{max} and p_{min} are identified. Next, we evaluate each slot $s \in p_{max}$ and calculate its potential

benefit if reassigned to p_{min}. The best slot $s^* \in p_{max}$ that maximizes balance improvement is selected and re-assigned from p_{max} to p_{min}. This correction process is repeated until the influence differences among all products are within θ. Finally, the algorithm outputs an allocation $(\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_\ell)$. This method guarantees feasibility by satisfying constraints, while achieving a near-optimal aggregated influence through greedy selection and local adjustments.

Algorithm 2: Greedy Heuristic with Balance Correction

Data: Billboard slots $\mathcal{BS}$, products $\mathcal{P} = \{p_1, p_2, \ldots, p_\ell\}$, budgets $\{k_1, \ldots, k_\ell\}$, threshold θ, error parameter ϵ
Result: Allocation $(\mathcal{S}_1, \ldots, \mathcal{S}_\ell)$ of slots to products

1 Initialize $\mathcal{S}_i \leftarrow \emptyset$ for all $i \in \mathcal{P}$, and $S^{'} \leftarrow \emptyset$ (set of used slots);
2 — **Greedy Allocation Phase** —;
3 **for** *each product* $i \in \mathcal{P}$ **do**
4 **while** $|\mathcal{S}_i| < k_i$ **do**
5 Set $k \leftarrow \max\{1, \lceil |\mathcal{BS}| \cdot 0.1 \rceil\}$;
6 Sample size: $r \leftarrow \left\lceil \frac{|\mathcal{BS}|}{k} \cdot \log \frac{1}{\epsilon} \right\rceil$;
7 Select candidate subset $\mathcal{R} \subseteq \mathcal{BS}$ uniformly at random, with $|\mathcal{R}| = \min(r, |\mathcal{BS}|)$;
8 **for** *slot* $s \in \mathcal{R}$ **do**
9 **if** $s \notin S^{'}$ *and* $s \notin \mathcal{S}_i$ *and cost*$(s) \leq k_i$ **then**
10 Compute marginal gain: $\Delta\mathcal{I}_i(s) = \mathcal{I}_i(\mathcal{S}_i \cup \{s\}) - \mathcal{I}_i(\mathcal{S}_i)$;
11 Choose $s^* = \arg\max_{s \in \mathcal{R}} \Delta\mathcal{I}_i(s)$;
12 **if** $s^* \neq \emptyset$ **then**
13 Assign slot: $\mathcal{S}_i \leftarrow \mathcal{S}_i \cup \{s^*\}$, $S^{'} \leftarrow S^{'} \cup \{s^*\}$;
14 **else**
15 break
16 — **Balance Correction Phase** —;
17 Compute $\mathcal{I}_i(\mathcal{S}_i)$ for all $i \in \mathcal{P}$;
18 **while** $\max_i \mathcal{I}_i(\mathcal{S}_i) - \min_j \mathcal{I}_j(\mathcal{S}_j) > \theta$ **do**
19 $p_{max} \leftarrow \arg\max_i \mathcal{I}_i(\mathcal{S}_i)$, $p_{min} \leftarrow \arg\min_i \mathcal{I}_i(\mathcal{S}_i)$;
20 **for** $s \in \mathcal{S}_{p_{max}}$ **do**
21 Compute gain if reassigned:
22 $gain_{min} = \mathcal{I}_{p_{min}}(\mathcal{S}_{p_{min}} \cup \{s\}) - \mathcal{I}_{p_{min}}(\mathcal{S}_{p_{min}})$;
23 $loss_{max} = \mathcal{I}_{p_{max}}(\mathcal{S}_{p_{max}}) - \mathcal{I}_{p_{max}}(\mathcal{S}_{p_{max}} \setminus \{s\})$;
24 $\Delta(s) = gain_{min} - loss_{max}$;
25 Select $s^* = \arg\max_s \Delta(s)$;
26 **if** $s^* \neq \emptyset$ **then**
27 Reassign: $\mathcal{S}_{p_{max}} \leftarrow \mathcal{S}_{p_{max}} \setminus \{s^*\}$, $\mathcal{S}_{p_{min}} \leftarrow \mathcal{S}_{p_{min}} \cup \{s^*\}$;
28 Update influences $\mathcal{I}_i(\mathcal{S}_i)$ for all i;
29 **else**
30 break
31 **return** $(\mathcal{S}_1, \ldots, \mathcal{S}_\ell)$;

Complexity Analysis. Now, we analyze the time and space requirements of Algorithm 2. In Line No. 1, initializing the allocation set will take $\mathcal{O}(\ell)$ time. In Line No. 3 `for loop` will execute for $\mathcal{O}(\ell)$ and the `while loop` will execute till the demand of each product is satisfied. So, in Line No. 3 to 15 in each iteration, computing the marginal gain for every feasible pair of product $i \in \mathcal{P}$

and available slot $s \in \mathcal{BS}$ will take $\mathcal{O}(\left\lceil \frac{|\mathcal{BS}|}{k} \cdot \log \frac{1}{\epsilon} \right\rceil \cdot \ell \cdot \bar{d})$ time per iteration. Since at most $\sum_{i=1}^{\ell} k_i$ slots are assigned, the total time requirements will be of $\mathcal{O}(\left\lceil \frac{|\mathcal{BS}|}{k} \cdot \log \frac{1}{\epsilon} \right\rceil \cdot \ell \cdot \bar{d} \cdot \sum_{i=1}^{\ell} k_i)$. Next, in Line No. 17 to 30 for each reassignment step requiring at most $\mathcal{O}(k_{\max})$ evaluations, where $k_{\max} = \max_i k_i$. Since this process may repeat for up to $\mathcal{O}(|\mathcal{BS}|)$ iterations, the overall cost of this phase is $\mathcal{O}(|\mathcal{BS}| \cdot k_{\max})$. So, combining both phases, the overall time complexity is $\mathcal{O}(\left\lceil \frac{|\mathcal{BS}|}{k} \cdot \log \frac{1}{\epsilon} \right\rceil \cdot \ell \cdot \bar{d} \cdot \sum_{i=1}^{\ell} k_i + |\mathcal{BS}| \cdot k_{\max})$. The additional space requirements for allocation set $(\mathcal{S}_1, \ldots, \mathcal{S}_\ell)$ will be $\mathcal{O}(|\mathcal{BS}|)$. The influence values $\mathcal{I}_i(\mathcal{S}_i)$ must be stored for each product takes $\mathcal{O}(\ell)$ space. So, the additional space requirement will be $\mathcal{O}(|\mathcal{BS}| + \ell)$. Hence, Theorem 3 holds.

Theorem 3. *The time and space requirements for Algorithm 2 are* $\mathcal{O}(\left\lceil \frac{|\mathcal{BS}|}{k} \cdot \log \frac{1}{\epsilon} \right\rceil \cdot \ell \cdot \bar{d} \cdot \sum_{i=1}^{\ell} k_i + |\mathcal{BS}| \cdot k_{\max})$ *and* $\mathcal{O}(|\mathcal{BS}| + \ell)$*, respectively.*

4 Experimental Setup

This section describes the experimental setup for the experiments. Initially, we start by reporting the datasets used in our experiments.

Dataset Description. We used two real-world datasets from New York City (NYC)[2] and Los Angeles (LA)[3], previously used in related studies [1,2,5,6]. The NYC dataset contains 227,428 check-ins (April 2012 to February 2013), and the LA dataset has 74,170 check-ins across 15 streets. Billboard data were obtained from LAMAR[4], yielding 1,031,040 slots for NYC and 2,135,520 slots for LA.

Key Parameters. All the parameters are summarized in Table 1. First, the Demand-Supply Ratio (α) defines the ratio $\alpha = \sigma^{\mathcal{P}}/\sigma^*$ as the proportion of global demand $\left(\sigma^{\mathcal{P}} = \sum_{i=1}^{n} \sigma_i\right)$ to total supply ($\sigma^* = |\mathcal{BS}|$). We evaluate α at four levels: $40\%, 60\%, 80\%, 100\%$. Second, the Average-Individual Demand Ratio (β) defines $\beta = \sigma^{\mathcal{P}''}/\sigma^*$ as the ratio of average individual demand to total supply, where $\sigma^{\mathcal{P}''} = \sigma^{\mathcal{P}}/|\mathcal{P}|$ denotes the average demand per product. The parameter β is used to scale the demand of individual products. Third, the demand for each product is computed as $\sigma = \lfloor \omega \cdot \sigma^* \cdot \beta \rfloor$, where $\omega \in [0.8, 1.2]$ is a random factor used to capture variations in product payments. Fourth, the threshold parameter (θ) controls fairness tolerance. It defines the maximum allowed difference between the highest and lowest product influence. Fifth, ϵ controls the probability of error in random sampling. We vary one parameter in each experiment and set the remaining parameters in their default settings (highlighted in bold). To solve the LP, we use the HiGHS[5] solver. All the Python codes are executed in HP Z4 workstations with 64 GB RAM and an Xeon(R) 3.50 GHz processor.

[2] https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page
[3] https://github.com/Ibtihal-Alablani
[4] http://www..lamar.com/InventoryBrowser
[5] https://highs.dev/

Table 1: Key Parameters

Parameter	Values
α	$40\%, 60\%, 80\%, \mathbf{100\%}$
β	$1\%, 2\%, \mathbf{5\%}, 10\%$
ϵ	$0.01, 0.05, \mathbf{0.1}, 0.2$
θ	$0.02, \mathbf{0.05}, 0.1, 0.2$
λ	$50m, \mathbf{100m}, 125m, 150m$

Baseline Methods.

Random + Balance Correction. In this approach, the billboard slots are selected uniformly at random to allocate to products until the budget constraint is satisfied. Next, the balance correction phase is applied, which ensures that the influence difference constraint is satisfied.

Top-k + Balance Correction. In this approach, the billboard slots are sorted according to their influence value. Then, sorted slots are assigned to the products individually until the budget constraint is satisfied. Finally, the balance correction is used to satisfy the influence difference.

Goals of our Experiments. In this study, we want to address the following Research Questions (RQ).

- **RQ1**: Varying α, β, how does the influence and runtime change?
- **RQ2**: Varying θ, how do the fairness gap and run time changes?
- **RQ3**: Varying ϵ and λ, how does the influence and run time change?

Experimental Results and Discussion

Budget, ϵ Vs. Influence. Figure 1(a,b,c,d) and Figure 2(a,b,c,d) show the effectiveness of the proposed solution approaches. We have four main observations. First, with increasing demand-supply ratio (α) from 40% to 100% with a fixed value of β, ϵ, and θ, the influence of all proposed and baseline methods increases. Second, the proposed 'LP+RR' approach achieves more influence than the 'Greedy Heuristic'. The 'LP+RR' approach gives better quality results (higher influence, tighter fairness balance) because it leverages LP relaxation, which approximates the global optimum before rounding. Third, among the baselines, the 'Top-k' approach performs better than 'Random' in terms of influence maximization. Fourth, the'Greedy Heuristic' is faster and scalable, however, at the cost of lower influence compared to 'LP+RR'. Figures 1(g) and 2(g) show that with increasing the error probability parameter (ϵ) value from 0.01 to 0.2, the quality of the solution degrades. The lower ϵ improves the quality of the solutions; however, the computational cost increases.

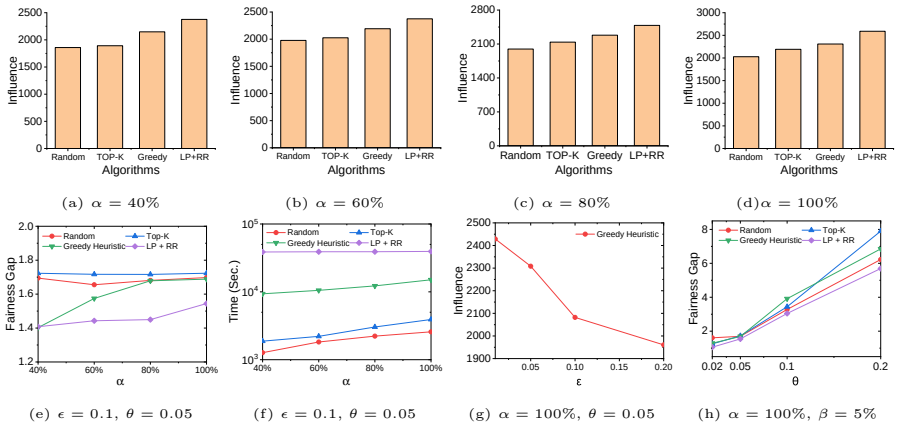

Fig. 1: Algorithms Vs. Influence (a, b, c, d), Varying α Vs. Fairness Gap (e), Time (f), Varying ϵ Vs. Influence (g), Varying θ Vs. Fairness Gap (h) with $|\mathcal{P}| = 20$, $\beta = 5\%$ for NYC dataset

θ Vs. Fairness Gap. Figures 1 (h) and 2 (h) show the effectiveness of the fairness tolerance parameter θ in maximizing influence across trajectories. The smaller θ shows that the difference in influence between the products is very close (strict balance), and the larger θ allows for larger gaps in influence between the products. With an increase in θ from 0.02 to 0.2, the influence gap for all proposed and baseline methods increases. Figures 1 (e) and 2 (e) show that with increasing α, the fairness gap also increases due to increasing the budget for the products. The 'Top-k' achieves the highest influence gap; however, 'LP + RR' achieves the lowest influence gap. If the trajectories show a dense overlap, some products will naturally have more influence than others, so we can allow more imbalance (bigger θ). For example, we say that the imbalance between products should not exceed, say, 0.1 or 10% of the typical coverage per slot.

Efficiency Test. Figure 1(f) and Figure 2(f) show the efficiency of the proposed and baseline methods, and from this, we have three main observations. First, the run time of the 'LP + RR' approach is higher compared to the 'Greedy Heuristic' because solving the LP requires huge computational time. Further, randomized rounding, budget repair, and balance repair take additional computational time. The 'Greedy Heuristic' takes less computational time because the sampling-based slot assignment reduces computational time. Second, with the increase of the demand-supply ratio (α), the run time for all the proposed and baseline methods increases because increasing α increases the demand for the slots of the products. Third, among baseline methods, 'Top-k' takes a higher run time than 'Random' because, before assigning slots to the products, the slots are sorted in descending order of their individual influence value.

Scalability Test. To show the scalability of the proposed approaches, we vary the number of products. We observe that the efficiency is very sensitive in 'LP+RR'

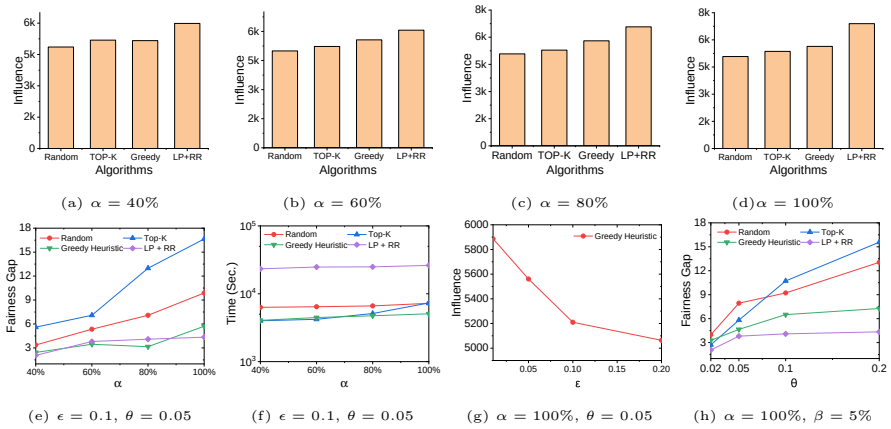

Fig. 2: Algorithms Vs. Influence (a, b, c, d), Varying α Vs. Fairness Gap (e), Time (f), Varying ϵ Vs. Influence (g), Varying θ Vs. Fairness Gap (h) with $|\mathcal{P}| = 20$, $\beta = 5\%$ for LA dataset

compared to the 'Greedy Heuristic' when we vary the number of products from 5 to 100. It is observed that the runtime increase in 'LP+RR' compared to 'Greedy Heuristic' and baselines is almost 60% to 80% and 90% to 95%.

Additional Discussions. Now, we discuss the impacts of additional parameters, i.e., varying distance (λ), Varying trajectory size, varying individual demand supply ratio (β), etc. First, with the increase in the distance over which billboard slots can influence the trajectories, the influence of the products increases. Second, with the increase in trajectory size, the influence of all the proposed and baseline methods increases because one slot can influence a large number of trajectories. Third, with increasing individual demand supply ratio β, the demand for the number of slots for each product increases, and for this reason, the computational time for all proposed and baseline methods increases. In all our experiments, we set the value of α, β, ϵ, θ, λ, and the trajectory size as 80%, 5%, 0.1, 0.05, 100 meters, and $120k$ (NYC), respectively, as a default setting.

5 Concluding Remarks

In this paper, we have introduced and studied the Multi-Product Influence Maximization Problem for the Balanced Popularity Problem. We show that the problem is NP-hard to solve optimally. We formulate this problem as a linear programming problem and use linear programming relaxation with randomized rounding. Further, we propose a greedy-based heuristic with balance correction to solve this problem. All the methods have been analyzed to understand their time and space requirements. They have been implemented with real-world datasets and compared with baseline methods to show their effectiveness and efficiency. However, the optimality has not been guaranteed. Now, our future work on this problem will remain concentrated on extending the methodologies for multiple advertisers.

References

1. Ali, D., Banerjee, S., Prasad, Y.: Influential billboard slot selection using pruned submodularity graph. In: International Conference on Advanced Data Mining and Applications. pp. 216–230. Springer (2022)
2. Ali, D., Banerjee, S., Prasad, Y.: Influential billboard slot selection using spatial clustering and pruned submodularity graph. arXiv preprint arXiv:2305.08949 (2023)
3. Ali, D., Banerjee, S., Prasad, Y.: Influential billboard slot selection under zonal influence constraint. In: European Conference on Advances in Databases and Information Systems. pp. 93–106. Springer (2024)
4. Ali, D., Banerjee, S., Prasad, Y.: Minimizing regret in billboard advertisement under zonal influence constraint. arXiv preprint arXiv:2402.01294 (2024)
5. Ali, D., Banerjee, S., Prasad, Y.: Regret minimization in billboard advertisement under zonal influence constraint. In: Proceedings of the 39th ACM/SIGAPP Symposium on Applied Computing. pp. 329–336 (2024)
6. Ali, D., Gupta, T., Banerjee, S., Prasad, Y.: Influential slot and tag selection in billboard advertisement. arXiv preprint arXiv:2401.10601 (2024)
7. Ali, D., Kumar, H., Banerjee, S., Prasad, Y.: An effective tag assignment approach for billboard advertisement. arXiv preprint arXiv:2409.02455 (2024)
8. Aslay, C., Bonchi, W.L.F., Goyal, A., Lakshmanan, L.V.: Viral marketing meets social advertising: Ad allocation with minimum regret. Proceedings of the VLDB Endowment **8**(7) (2015)
9. Balas, E., Padberg, M.W.: On the set-covering problem. Operations Research **20**(6), 1152–1161 (1972)
10. Barahona, F., Chudak, F.A.: Near-optimal solutions to large-scale facility location problems. Discrete Optimization **2**(1), 35–50 (2005)
11. Ke, X., Khan, A., Cong, G.: Finding seeds and relevant tags jointly: For targeted influence maximization in social networks. In: Proceedings of the 2018 international conference on management of data. pp. 1097–1111 (2018)
12. Raghavan, P., Tompson, C.D.: Randomized rounding: a technique for provably good algorithms and algorithmic proofs. Combinatorica **7**(4), 365–374 (1987)
13. Wang, L., Yu, Z., Yang, D., Ma, H., Sheng, H.: Efficiently targeted billboard advertising using crowdsensing vehicle trajectory data. IEEE transactions on industrial informatics **16**(2), 1058–1066 (2019)
14. Zhang, P., Bao, Z., Li, Y., Li, G., Zhang, Y., Peng, Z.: Trajectory-driven influential billboard placement. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining. pp. 2748–2757 (2018)
15. Zhang, P., Bao, Z., Li, Y., Li, G., Zhang, Y., Peng, Z.: Towards an optimal outdoor advertising placement: When a budget constraint meets moving trajectories. ACM Transactions on Knowledge Discovery from Data (TKDD) **14**(5), 1–32 (2020)
16. ZHANG, Y., LI, Y., BAO, Z., MO, S., ZHANG, P.: Optimizing impression counts for outdoor advertising.(2019). In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, Alaska. pp. 4–8 (2019)
17. Zhang, Y., Li, Y., Bao, Z., Zheng, B., Jagadish, H.: Minimizing the regret of an influence provider. In: Proceedings of the 2021 International Conference on Management of Data. pp. 2115–2127 (2021)