PLUTUS: Leveraging Location-based Social Networks to Recommend Potential Customers to Venues

Mohamed Sarwat Ahmed Eldawy Mohamed F. Mokbel John Riedl
Department of Computer Science and Engineering
University of Minnesota
200 SE Union Street, Minneapolis, MN 55455, USA
sarwat@cs.umn.edu, eldawy@cs.umn.edu, mokbel@cs.umn.edu, riedl@cs.umn.edu

***Abstract*—In a business setting, the customer value is crucial as it determines how much it is worth spending to acquire a particular customer. Viral marketing techniques leverages social ties among users to help advertising a particular product. Recently, as mobile devices (e.g., smart phones, GPS devices) became ubiquitous, location-based social networking websites (e.g., Gowalla, BrightKite, Foursquare) are getting more and more popular. Along with location-based social networking services being prominent, new kind of data came into play besides the traditional social networking data: (1) *Spatial data*: represents the users geo-locations, venues geo-locations and information about users visiting different venues. (2) *Users Opinions data*: represents how much a user likes the venues she visits (e.g., Alice visited restaurant A and gave it a rating of five over five). In this paper, we present PLUTUS; a framework that assists venues (e.g., restaurant, gym, shopping mall) owners in growing their business. To recommend the best set of customers, PLUTUS takes three main aspects into consideration: (1) Social aspect, (2) Spatial aspect, and (3) Users opinions aspect. To this end, PLUTUS proposes two main algorithms: (1) *Profit Calculation*: It is responsible of calculating the total profit that a user u may add to a venue v taking into account the social, spatial, and user opinions aspects. (2) *Profit Maximization*: This algorithm is used to maximize the total profit of a given venue. We evaluated PLUTUS using real data set extracted from an existing Location-based Social Networking website, *Foursquare*. The results show that *Plutus* achieves higher estimated profit and more efficient profit calculation than naive marketing algorithms.**

I. INTRODUCTION

In a business setting, the customer value is crucial as it determines how much it is worth spending to acquire a particular customer [1]. Past work [2], [3], [4] has proposed using social ties to help advertising a particular product; a technique also known as *viral marketing*. By using social ties, the customer value is not only determined by how much dollars this particular customer is willing to spend to buy the advertised product. However, the customer value is also determined by how much influence s/he has on his/her friends (i.e., *network value*), as a more influential customer might influence his/her friends to buy the product, as well.

Recently, as mobile devices (e.g., smart phones, GPS devices) became ubiquitous, location-based social networking services (e.g., BrightKite [1], Foursquare [2], Gowalla [3], and Facebook Places) are getting more and more popular. For instance, as of September, 2012, Foursquare claims to have over 25 million people worldwide, and over billions of check-ins with millions more every day [5]. Users, in a location-based social networking website, can select their friend list as well as getting listed as friends to other users in a same way like traditional social networking systems. In addition, users are associated with a geo-location, and might alert friends when visiting a venue (e.g., restaurant, bar) by checking-in on their mobile phones (e.g., iPhone, Android).

Along with location-based social networking services being ubiquitous, new kind of data came into play besides the traditional social networking (i.e., friendship) data: (1) *Spatial data*: represents the users geo-locations, venues (e.g., restaurant, gym, shopping mall) geo-locations and information about users visiting different venues. (2) *Users Opinions data*: represents how much a user likes the venues she visits (e.g., Alice visited restaurant A and gave it a rating of five over five). The combination of *social, spatial, and opinions* data extracted from location-based social networking services can be leveraged to help venues grow their business. This can be achieved by recommending a set of users $\mathcal{U}$ to a venue v who are expected to influence other users that are: (a) socially close to users in $\mathcal{U}$, (b) nearby venue v geo-location, and (c) are expected to like venue v.

In this paper, we present PLUTUS; a framework that assists venues owners in growing their business. The framework helps marketers efficiently spending their budget in a way that maximizes the expected venue's profit. For instance, if the business owner has to distribute $\mathcal{K}$ coupons, PLUTUS helps her selecting the best $\mathcal{K}$ customers that maximizes the expected profit. For simplicity, we measure the expected profit in terms of the number of people visiting the venue per a period of time. For example, If venue A is visited by 1000 users in 10 days and venue B is visited by 600 users in the same 10 days, implying that venue A has higher profit than B.

To recommend the set of users that leads to achieving higher profit, PLUTUS takes into account three main aspects:

[1]BrightKite: http://brightkite.com
[2]Foursquare: http://foursquare.com
[3]Gowalla: http://gowalla.com.

Survey Item	Mean	Stdev
Would you consider visiting a restaurant, if you know that a friend of yours likes and visits this restaurant	0.96	0.2
If you know that your favorite celebrity likes and visits a particular restaurant, will this encourage you to visit this restaurant	0.81	0.43
If you know that a professional food critique gave a high rating to some restaurant, will this encourage you to visit this restaurant	0.74	0.39
Do you use the "check-in" (location sharing) option in any social networking website	0.62	0.48

TABLE I. MECHANICAL TURK SURVEY

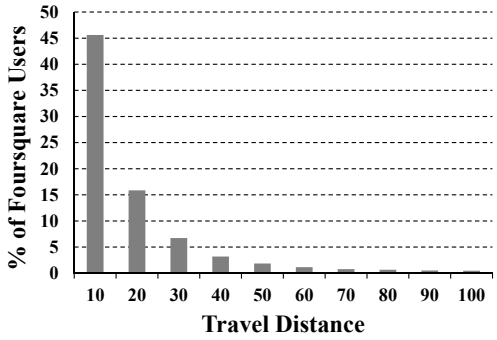


Fig. 1. Foursquare Users Travel Distance

(1) *Social Network*: The framework leverages the social ties among users drawn from the social graph, and hence acquires the most influential users in the social graph. In other words, PLUTUS keeps track of how each user influences each of his friends in the social graph and uses that in recommending the best users. (2) *Spatial*: PLUTUS considers the venue location (i.e., address), the user location, and user's check-ins (i.e., visits) at different venues to retrieve users that are spatially relevant to the business. (3) *Users Opinions*: The framework takes into account the user ratings to different venues to acquire the users that are expected to like the venue.

To this end, PLUTUS proposes two main algorithms: (1) *Profit Calculation*: It is responsible for calculating the total profit that a user u may add to a venue v taking into account the social, spatial, and user opinions aspects. (2) *Profit Maximization*: This algorithm has two versions (*Celebrity-based* and *Coupon-based* profit maximization) and each of them is used to maximize the total profit of a given venue. Each version, more or less, serves the same purpose, i.e., maximizing the expected venue profit (i.e., increase the number of people visiting the place). However, they differ based upon the user requirements and expectations; A PLUTUS user may select one of the following versions of the profit maximization algorithm: (a) *Celebrity-based* Profit Maximization: Given a total marketing budget (e.g., total amount of dollars dedicated for the marketing campaign), PLUTUS recommends any number of users that maximizes the total profit of the designated venue. (b) *Coupon-based* Profit Maximization: Given a maximum marketing budget per user (e.g., coupon price), PLUTUS recommends $\mathcal{K}$ users that maximizes the expected total profit of the designated venue.

In the rest of the paper, we first present an initial study in section II to motivate the problem. we then give an overview of PLUTUS in section III. We then describe the *profit calculation* algorithm in section IV. Both versions of the *Profit maximization* algorithm are then explained in section V. Section VI evaluate the performance of PLUTUS. Related works are highlighted in section VII. Finally, Section VIII summarizes PLUTUS contributions and concludes the paper.

II. INITIAL STUDY

Mechanical Turk Study. To motivate our work, we ran a user study on Amazon Mechanical Turk [6]. We conducted a survey on a hundred users through Amazon Mechanical Turk. The main aim of the survey was to quantitatively measure whether users are influenced by their social ties (i.e., friends, favorite celebrities, and experts) when it comes to visiting venues (e.g., restaurants). We have asked the users the questions shown in table I. In the survey, 62 participants claims that they use the check-in option provided by some social networking services (e.g., Facebook, Foursquare). More importantly, 96 participants admit they would go to a restaurant if they know their friends likes and visits this restaurant. On the other hand, 81 of participants said they would consider visiting a restaurant if they knew their favorite celebrity prefers and visits that restaurant. Finally, 74 mentioned they would visit a restaurant if they knew that an expert food critique gave a high rating to that restaurant. The take-away from the study is that users are substantially influenced by the opinions of their acquaintances, celebrities, and experts when visiting restaurants. Moreover, the numbers in Table I show that users are influenced by their friends more than experts and celebrities. A more profound user study is performed by Lindqvist et.al in [7].

Foursquare Data Analysis. We analyzed data from the Foursquare location-based social network containing user visits data for 1M users to 643K venues across the United States. Figure 1 gives the travel distance histogram of the examined Foursquare users. We figured out that users tend to travel a limited distance when visiting these venues. We refer to this property as travel locality. In our analysis of Foursquare data, we observed that 45% of users travel 10 miles or less, while 75% travel 50 miles or less. The results are similar to the results of location-based social network data analysis given in [8]. This observation suggests that customers closer in travel distance to a venue are expected to bring higher profit to that venue. In other words, the farther a customer is from a venue, the more that customer loses her added value to that venue.

III. PLUTUS OVERVIEW

Figure 2 gives an overview of PLUTUS. As presented in the figure, PLUTUS takes three kinds of data, as input: (1) Social Network Data, (2) Spatial Data, and (3) Users Opinions data, described as follows.

A. Social Networking Data

The social networking data model is a typical directed social graph in which nodes represent the set of users $\mathcal{U} = \{u_1, u_2, ..., u_n\}$, and a set of edges $\mathcal{E} = \{e_1, e_2, ..., e_m\}$ represents the friendship relationship between users. In the rest of the

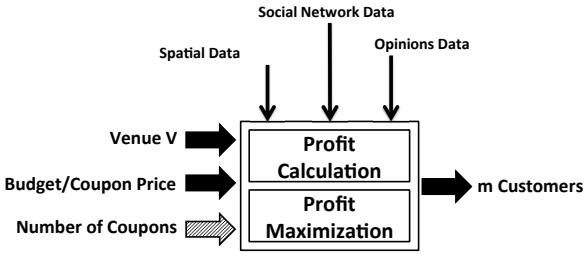


Fig. 2. PLUTUS framework overview

paper, we use the notions *user* and *node* interchangeably to represent a social graph node. Each user u_i is associated with a static profit $\mathcal{S}_{u_i,v_j}$ which represents the expected static profit gained when *Customer* (i.e., user) u_i visits the venue v_j. For example, when $Alice$ visits restaurant R, she may have a 12 dollars meal, so $\mathcal{S}_{Alice,R}$ is equal to \$12. Each user has a cost $\mathcal{C}_{u_i,v_j}$ defined as the amount of dollars needed to attract a *Customer* (i.e., user) u_i to visit the venue v_j. For instance, the cost to make *Justin Bieber* visit Gym X, $\mathcal{C}_{JustinBieber,X}$ is one million dollars. Each directed edge $e \in \mathcal{E}$ between two users u_1 and u_2 is annotated with the influence that u_1 has on u_2, denoted as $\mathcal{I}_{u_i,u_l}$, and formally defined as follows:

Definition 1. *Given a user u_1 and another user u_2, $\mathcal{I}_{u_1,u_2}$ is defined as the influence of user u_1 on u_2. It is the probability that user u_2 visits a place given that user u_1 already visited the same place. Notice that $\mathcal{I}_{u_1,u_2}$ is not a symmetric relation. In other words, the value of $\mathcal{I}_{u_1,u_2}$ may be different from the value of $\mathcal{I}_{u_2,u_1}$.*

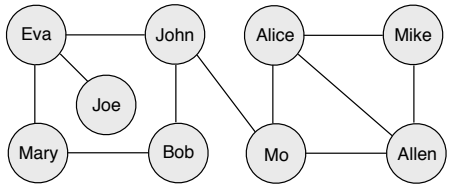
For instance in Figure 3, $\mathcal{I}_{Eva,Joe}$ is equal to 0.3 and $\mathcal{I}_{Joe,Eva}$ is equal to 0.7. That means that Joe has higher influence on Eva than Eva has on him.

B. Spatial Data

The spatial data gives information on the user location, venue location and the visits of different users to different venues. Each User $u_i \in \mathcal{U}$ is associated with a geo-location (i.e., Longitude and Latitude) $\mathcal{L}_{u_i}$. For instance, if the address of user $_{u_i}$ is $\langle x_1,y_1 \rangle$, then $\mathcal{L}_{u_i}$=$\langle x_1,y_1 \rangle$. Similarly, each venue $v_i \in \mathcal{V}$ has a geo-location $\mathcal{L}_{v_i}$ referring to its spatial address. For instance, if the address of restaurant $\mathcal{R}$ is $\langle x_1,y_1 \rangle$, then $\mathcal{L}_R$=$\langle x_1,y_1 \rangle$. Each user might have visited different venues at different times, by means of the *check-in* function available in location-based social networks. We have a set of check-ins $\mathcal{C}$ = $\{\langle u_1,v_1,t_1 \rangle, \langle u_2,v_2,t_2 \rangle, ..., \langle u_i,v_j,t_l \rangle\}$. A check-in c = $\langle u_i,v_j,t_l \rangle$, which means that user u_i has visited venue v_j at time t_l.

C. Users Opinions Data

The opinions data define how much a user u likes a venue v. The users opinions can be extracted from the ratings they gave to different venues they visited before. However, in case the user did not visit the venue before, we employ a rating prediction technique to predict how the user might have rated that particular venue. A well-know method for rating prediction is *collaborative filtering* [9], [10]. Collaborative



U1	U2	Influence
Eva	Joe	0.3
Eva	Mary	0.1
Eva	John	0.3
Joe	Eva	0.7
Mary	Eva	0.2
Mary	Bob	0.3
Bob	Mary	0.4
Bob	John	0.1
John	Bob	0.4
John	Eva	0.2
John	Mo	0.7

User	C	S	R	Location
Eva	5	100	1.8	New York
Mary	9	30	4.2	New York
Joe	1	80	5.0	New York
John	5	10	4.0	St. Paul
Bob	50	1	2.5	Minneapolis
Mo	100	6	1.0	Minneapolis
Allen	10	5	3.5	San Diego
Mike	100	60	3.3	Minneapolis
Alice	5	8	5.0	San Francisco

Fig. 3. PLUTUS Data Model

filtering assumes a set of n users $\mathcal{U} = \{u_1, ..., u_n\}$ and a set of m venues $\mathcal{V} = \{v_1, ..., v_m\}$. Each user u_j expresses opinions about a set of venues $\mathcal{V}_{u_j} \subseteq \mathcal{V}$. For a venue v, PLUTUS calculates the predicted rating that u would give to venue v using item-based collaborative filtering technique (see section IV-A for details).

D. Plutus Functionality

Input and Output Parameters. PLUTUS takes up to three input parameters from its users, as shown in Figure 2. These parameter are as follows:

1) Venue V: It is the venue for which PLUTUS will recommend a set of potential customers that are expected to increase the profit of that venue.
2) Budget/Coupon price: It is a mandatory parameter used to determine the amount of dollars dedicated to the marketing campaign. This amount may represent a full budget or a single coupon price; PLUTUS deals with both scenarios.
3) Number of customers (i.e.,) $\mathcal{K}$: It is an optional parameter used to determine the numbers of customers to be recommended to the designated venue V to increase its expected profit.

The output of PLUTUS is a set of customers (i.e., social network users) that are expected to increase the profit of the input venue V. The framework may return a set of customers with aribitrary size, unless $\mathcal{K}$ is given as input. In this case, PLUTUS returns a number of customers m such that $m \leq \mathcal{K}$.

PLUTUS Algorithms. PLUTUS has two main algorithms: (1) *Profit Calculation Algorithm*: It is used to calculate the profit of recommending a user u to a venue v, and (2) *Profit Maximization algorithm*: Given the calculated profit, this algorithm determines what *customers* (i.e, users) should be recommended to venue V to increase its profit. The profit maximization algorithm has two versions: (a) *Celebrity-based* Profit Maximization: Given a total marketing budget (e.g., total amount of dollars dedicated for the marketing campaign)

Algorithm 1 Network Profit Calculation

1: **Function** CalculateNetworkProfit(Customer u_i, Venue v)
2: $NetworkProfit \leftarrow 0.0$
3: **for each** User $\mathcal{F}_{ui}^{j}$ **in** User u_i Friends List **do**
4: $NetworkProfit \leftarrow NetworkProfit +$ CalculateTotalProfit($\mathcal{P}_{\mathcal{F}_{ui}^{j}}$,$v$) $\times$ $\mathcal{I}_{u_i,\mathcal{F}_{ui}^{j}}$
5: **end for**
6: **return** $NetworkProfit$

provided by venue V, recommend a set of users that maximize the total profit. In this case, the *Budget/Coupon price* input parameter represents the whole budget. (b) *Coupon-based* Profit Maximization: Given a maximum marketing budget per user (e.g., coupon cost) provided by venue V, recommend $\mathcal{K}$ users that maximize the expected total profit. In such case, the *Budget/Coupon price* input parameter represents the coupon price. Moreover, the *Number of customers* parameter must be specified in the input. In the rest of the paper, we will describe each algorithm in details.

IV. Profit Calculation

The profit calculation algorithm estimates the profit of recommending *customer* u_i to venue v. $\forall\ u_i \in \mathcal{U}$, $\exists\ \mathcal{P}_{u_i,v}$ ($\mathcal{P}_{u_i,v} \geq 0$) such that $\mathcal{P}_{u_i,v}$ represents the total profit user u_i brings to venue v:

$$\mathcal{P}_{u_i,v} = \mathcal{N}_{u_i,v} + \mathcal{S}_{u_i,v} - \mathcal{C}_{u_i,v} \tag{1}$$

Both static profit $\mathcal{S}_{u_i,v}$ and Cost $\mathcal{C}_{u_i,v}$ has been explained before in section III. $\mathcal{N}_{u_i,v}$ represents the network profit which is formally defined as follows:

Definition 2. *The Network Profit $\mathcal{N}_{u_i,v} \geq 0$ is defined as the expected network profit gained when user u_i visits the venue v. It is dynamic in the sense that it depends on the social ties of user u_i. $\mathcal{N}_{u_i,v}$ is dependent on how u_i influences his/her friends, and how that influence spreads through the entire social graph. The network profit that venue v gains by acquiring user u_i is calculated as follows:*

$$\mathcal{N}_{u_i,v} = \sum_{j=0}^{f_{u_i}} \mathcal{P}_{\mathcal{F}_{ui}^{j},v} \times \mathcal{I}_{u_i,\mathcal{F}_{ui}^{j}} \tag{2}$$

Such that f_{u_i} represents the total number of friends of user u_i. $\mathcal{F}_{ui}^{j}$ denotes the j^{th} friend of user u_i and $\mathcal{P}_{\mathcal{F}_{ui}^{j},v}$ represents the total profit user $\mathcal{F}_{ui}^{j}$ brings to venue v.

Algorithm 1 gives the pseudocode of the network profit calculation algorithm that takes a customer u_i and a venue v as input, and returns the total expected network profit that venue v gains by acquiring customer u_i. The network profit $\mathcal{N}_{u_i,v}$ is calculated by running a breadth first search traversal over the social graph starting from user (node) u_i (lines 2 to 5). For each friend $\mathcal{F}_{ui}^{j}$ of user u_i, we calculate the total profit of $\mathcal{F}_{ui}^{j}$ by invoking the CalculateTotalProfit algorithm (explained later in section IV-C) and then multiplying it by $\mathcal{I}_{u_i,\mathcal{F}_{ui}^{j}}$; the influence of user u_i on his friend $\mathcal{F}_{ui}^{j}$. The algorithm goes for all social graph descendants of u_i, recursively calculate the network profit of each descendant, and adds its share in the network profit to the final network profit of user u_i; $NetworkProfit$.

A. Profit Calculation with Users Opinions

Notice that the total profit presented in equation 1 does not take the user rating (i.e., opinion) into account. The user rating to venue $\mathcal{R}_{u_i,v}$ is necessary to be included in the equation, as the user interest in the venue v incurs a high effect on the total profit. In other words, the total profit should be higher in the case of high user rating than that of low rating. The user u_i rating to venue v is incorporated in the equation, as follows:

$$\mathcal{P}^{'}_{u_i,v} = \mathcal{R}_{u_i,v} \times (\mathcal{N}_{u_i,v} + \mathcal{S}_{u_i,v} - \mathcal{C}_{u_i,v}) \tag{3}$$

As $\mathcal{R}_{u_i,v}$ takes a value from 0 to 1, it acts as a damping factor for the calculated profit. In other words, if $\mathcal{R}_{u_i,v}$ is equal to zero that means the user u_i does not like the venue v, and hence the profit that user u_i brings to venue v is equal to zero. On the hand, in case $\mathcal{R}_{u_i,v}$ is equal to one which means the user u_i likes the venue v, which affects the total profit positively. If a user u did not visit (rate) venue v before, PLUTUS thus predicts the value of $\mathcal{R}_{u_i,v}$ based on the history of available user opinions.

To this end, we build an item-based collaborative filtering model by analyzing the entire user/venue rating space, and using statistical techniques to find correlated venues. There are several methods to perform collaborative filtering including item-based [11], user-based [12], regression-based [11], or approaches that use more sophisticated probabilistic models (e.g., Bayesian Networks [13]). These correlations are measured by a *score*, or weight, that defines the strength of the relation. The item-item model builds, for each of the m venues $\mathcal{V}$ in the database, a list $\mathcal{L}$ of *similar* venues. Given two venues v_p and v_q, we can derive their similarity score $sim(v_p, v_q)$ by representing each as a vector in the user-rating space, and then use a similarity function over the two vectors to compute a numeric value representing the strength of their relationship.

The similarity function, $sim(v_p, v_q)$, computes the similarity of vectors v_p and v_q using *only* their co-rated dimensions. Finally, we store v_p, v_q, and $sim(i_p, i_q)$ in the model. The similarity measure need not be symmetric, i.e., it is possible that $sim(i_p, i_q) \neq sim(v_q, v_p)$. Many similarity measures have been proposed in the literature [14], [11], among which we have selected Pearson Correlation similarity measure as it is one of the most widely used in literature. it measures the similarity between venues using their Pearson correlation coefficient as follows:

$$sim(v_p, v_q) = \frac{\sum_{u \in \mathcal{U}_c}(r_{u,v_p} - \overline{r}_{v_p})(r_{u,v_q} - \overline{r}_{v_q})}{\sigma_{v_p}\sigma_{v_q}} \tag{4}$$

$\mathcal{U}_c$ represents users who co-rated items v_p and v_q, r_{u,v_p} and r_{u,v_q} represent a user's ratings, and $\overline{r}_{v_p}$ and $\overline{r}_{v_q}$ represent the average rating for venues v_p and v_q, respectively. σ_{v_p} and σ_{v_q} are the standard deviations for v_p and v_q

We then use the model to predict ratings for venues that a user u_a has not rated. Rating predictions are produced by performing aggregation over the generated collaborative filtering model. Using the item-based collaborative filtering technique, we compute the predicted rating $\mathcal{R}_{u_i,v}$ for venue v and user u_i as a weighted sum [11]:

$$\mathcal{R}_{u_i,v} = \frac{\sum_{l \in \mathcal{L}} sim(v,l) * r_{u_i,l}}{\sum_{l \in \mathcal{L}} sim(v,l)} \tag{5}$$

The prediction is the sum of the user's rating for a related venue l, $r_{u_i,l}$, weighted by the similarity to the candidate item v. The prediction is normalized by the sum of scores between v and l.

B. Profit Calculation with Travel Penalty

The idea is to exploit the observation that users limit their choice of spatial venues based on travel distance (mentioned in section II). *Travel penalty* is a value that penalizes the total profit of a user based on her distance from the designated venue. *Travel penalty* may incur expensive computational overhead by calculating travel distance to each user. Thus, we employ an efficient query processing technique capable of *early termination* to the total profit without calculating the travel distance to all users. The total profit, after incorporating the spatial effect, is expressed as follows:

$$\mathcal{P}^{''}_{u_i,v} = \mathcal{T}_{u_i,v} \times \mathcal{R}_{u_i,v} \times (\mathcal{N}_{u_i,v} + \mathcal{S}_{u_i,v} - \mathcal{C}_{u_i,v}) \quad (6)$$

$\mathcal{T}_{u_i,v}$ is the *travel penalty* applied due to the euclidean distance between user u_i and venue v. Notive that $\mathcal{T}_{u_i,v}$ takes a value between zero and one. When ($\mathcal{T}_{u_i,v}$ = 0), this means that the distance between u_i and v is too high, and ($\mathcal{T}_{u_i,v}$ = 1) means that u_i is too close to v. To add the geo-spatial effect, we multiply the *travel penalty* $\mathcal{T}_{u_i,v}$ by the total profit value in equation 3. The final expected total profit that user u_i brings to venue v is given in equation 6.

C. Profit Calculation Algorithm

From equations 1 to 6, the total profit in PLUTUS is calculated as follows:

$$\mathcal{P}^{''}_{u_i,v} = \mathcal{T}_{u_i,v} \times \mathcal{R}_{u_i,v} \times (\sum_{j=0}^{f_{u_i}} \mathcal{P}_{\mathcal{F}^j_{ui},v} \times \mathcal{I}_{u_i,\mathcal{F}^j_{ui}} + \mathcal{S}_{u_i,v} - \mathcal{C}_{u_i,v})$$

Algorithm 2 provides the pseudocode for the total profit calculation algorithm. The algorithm takes a customer u_i and a venue v as input, and returns the total expected profit that venue v gains by acquiring customer u_i. First, the algorithm retrieves the user static profit $\mathcal{S}_{u_i,v}$ and Static cost $\mathcal{C}_{u_i,v}$ necessary for user u_i to visit venue v. Notice that both $\mathcal{S}_{u_i,v}$ and $\mathcal{C}_{u_i,v}$ are saved for each user in the social graph (see section III-A). The algorithm then retrieves the rating score that user u_i gave to venue v. If user u_i did not rate venue v before, then equation 5 is applied to predict the value of $\mathcal{R}_{u_i,v}$. The algorithm also calculates the travel penalty $\mathcal{T}_{u_i,v}$ based on the distance of u_i from v. Then, we calculate the total profit by subtracting $\mathcal{C}_{u_i,v}$ from $\mathcal{S}_{u_i,v}$ and then adding the network profit calculated using the CalculateNetworkProfit algorithm. Finally, the returned value is multiplied by both $\mathcal{T}_{u_i,v}$ and $\mathcal{R}_{u_i,v}$ to incorporate both the spatial effect and the user opinions effect in the profit calculation. The final returned value $TotalProfit$ is the expected profit $\mathcal{P}^{''}_{u_i,v}$ that user u_i brings to venue v.

Algorithm 2 Plutus Profit Calculation

```
1:  Function CalculateTotalProfit(Customer u_i, Venue v)
2:  CurrentNode ← Retrieve the graph node that represents u_i
3:  S_{u_i,v} ← GetStaticProfit(u_i,v)
4:  C_{u_i,v} ← GetStaticCost(u_i,v)
5:  R_{u_i,v} ← GetRating(u_i,v) /* using equation 5 */
6:  T_{u_i,v} ← GetTravelPenaly(u_i,v)
7:  if CurrentNode.isVisited equals true then
8:      TotalProfit ← 0.0
9:  else
10:     CurrentNode.isVisited ← true
11:     TotalProfit ← T_{u_i,v} × R_{u_i,v} × (S_{u_i,v} − C_{u_i,v} +
                      CalculateNetworkProfit(u_i,v))
12: end if
13: return TotalProfit
```

V. PROFIT MAXIMIZATION

The profit maximization algorithm retrieves the set of *customers* that are expected to maximize the total venue profit. PLUTUS supports two versions of the profit maximization algorithm: (1) Celebrity-based profit maximization (section V-A), and (2) Coupon-based profit maximization (section V-B). In the rest of this section, we explain both versions in details.

A. Celebrity-based Profit Maximization

The celebrity-based profit maximization algorithm allows the venue (e.g., restaurant) owners to specify a total budget they are planning to spend on the marketing campaign. In this case, PLUTUS retrieves a set of customers (i.e., users in the social graph) that are expected to increase the profit for the designated venue. The celebrity-based profit maximization problem can be formulated as an integer linear program, as follows:

$$\begin{array}{ll} \max & \sum_{i=0}^{n} x_i \times \mathcal{P}^{''}_{u_i,v} \\ \text{s.t.} & \sum_{i=0}^{n} x_i \times \mathcal{C}_{u_i,v} \leq \mathcal{B}_v \\ & x_i \in \{0,1\}. \end{array}$$

Assume the social graph described in section III such that n is the total number of nodes (i.e., customers) in the graph. Let $\mathcal{P}^{''}_{u_i,v}$ denote the total profit that results from incorporating customer u_i in the result set of venue v (calculated using Algorithm 2) and let $\mathcal{B}_v$ represent the total budget specified by venue v owner. Let x_i be set to 1 if user u_i is selected and reset to 0 otherwise. We need to maximize the total profit, such that the total cost of all selected *customers* does not exceed the total budget $\mathcal{B}_v$.

The celebrity-based profit maximization problem is *NP-Complete*; 0/1 knapsack problem can be reduced to our problem (proof omitted for brevity). The problem is even harder than the 0/1 knapsack problem; as the value $\mathcal{P}^{''}_{u_i,v}$ is not constant. The order in which we pick *customers* leads to different profit values, $\mathcal{P}^{''}_{u_i,v}$. Moreover, the problem at hand may be so hard that it is likely that a c-approximation solution does not exist. Hence, we resort to a simple strategy such as greedy, which at least has the desirable property that it is fast to execute and intuitive to understand.

Algorithm. The greedy hill climbing algorithm at each iteration first computes a set of m candidate Customers $A \subseteq \mathcal{U}$, such that the cost of adding each customer a_i, $\mathcal{C}_{a_i,v}$, to the result set does not exceed the total venue budget $\mathcal{B}_v$. Therefore, we will pick the customer that leads to the highest profit value $\mathcal{P}^{''}_{u_i,v}$, exclude it from being picked in the future, and update the capacity usage of the total venue budget $\mathcal{B}_v$. The algorithm terminates when the remaining budget in the total venue budget $\mathcal{B}_v$ is less than or equal zero or there is no customers in the candidate set A. The pseudocode for the celebrity-based hill climbing algorithm is omitted for brevity.

Limited Customers. An extension of the celebrity-based profit maximization problem is to limit the number of customers to allow the framework users to specify a total budget and a fixed number of required customers $\mathcal{K}$. In this case, PLUTUS retrieves a set of $\mathcal{K}$ customers (i.e., users in the social graph) that are expected to increase the venue profit. In such case, the problem can be formulated similarly to the celebrity-based profit maximization problem, with an extra constraint as follows:

$$\begin{array}{rl} \max & \sum_{i=0}^{n} x_i \times \mathcal{P}^{''}_{u_i,v} \\ \text{s.t.} & \sum_{i=0}^{n} x_i \times \mathcal{C}_{u_i,v} \leq \mathcal{B}_v \\ & \sum_{i=0}^{n} x_i <= \mathcal{K} \\ & x_i \in \{0,1\}. \end{array}$$

The additional constraint is necessary so that we ensure that only $\mathcal{K}$ Customers are returned in the result set. The algorithm to solve that problem is similar to the celebrity-based hill climbing algorithm, except that the termination condition is extended to incorporate the the number of customers limit.

B. Coupon-based Profit Maximization

The coupon-based profit maximization version allows venues owners to specify a set of $\mathcal{K}$ coupons and a fixed price $\mathcal{B}_v$ for each coupon. The framework will then fetch the venue a set of customers (i.e., users in the social graph) that are expected to increase the profit. The Coupon scheme can be formulated as follows:

$$\begin{array}{rl} \max & \sum_{i=0}^{n} x_i \times \mathcal{P}^{''}_{u_i,v} \\ \text{s.t.} & x_i \times \mathcal{C}_{u_i,v} \leq \mathcal{B}_v \\ & \sum_{i=0}^{n} x_i <= \mathcal{K} \\ & x_i \in \{0,1\}. \end{array}$$

Example. A restaurant owner is willing to distribute 10 coupons, each worth a 100 dollars value of meals purchases. Hence, the restaurant owner specifies $\mathcal{K}$=10 and $\mathcal{B}_v$=100. PLUTUS therefore selects a set of m (i.e., $m \leq \mathcal{K}$) users that are expected to maximize the total restaurant profit.

Similar to the celebrity-based hill profit maximization, we use a greedy strategy to select m customers, such that $m \leq \mathcal{K}$. The algorithm follows the same steps as in the celebrity scheme algorithm, except that at each iteration it picks a customer a_i that leads to the highest profit value $\mathcal{P}^{''}_{u_i,v}$ for venue v whereas the cost $\mathcal{C}_{a_i,v}$ does not exceed the coupon price $\mathcal{B}_v$. The algorithm terminates when the remaining number of coupons reaches zero, which means all coupons are taken.

PLUTUS Coupon-based Algorithm. In the algorithm, we aim to avoid calculating $\mathcal{P}^{''}_{u_i,v}$ in Equation 6 for *all* customers in the social graph to find the k customers to offer them coupons, which can become quite expensive given the need to compute the network profit $\mathcal{N}_{u_i,v}$ for each customer. To avoid such computation, we evaluate customers in monotonically increasing order of travel penalty (mentioned earlier in section IV-B), enabling us to use early termination principles from top-k query processing [15], [16], [17].

Algorithm 3 provides the pseudocode of the PLUTUS coupon-based profit maximization algorithm that takes a venue v, a coupon price $\mathcal{B}_v$, and coupons count $\mathcal{K}$ as input, and returns the list R of m ($m \leq \mathcal{K}$) customers that are expected to maximize the total profit. The algorithm starts by running a k-nearest-neighbor algorithm to populate the list R with k customers with lowest travel penalty while the cost to acquire each of these customer $\mathcal{C}_{u_i,v}$ is less than or equal to the coupon price $\mathcal{B}_v$. The list R is sorted by the the total computed profit $\mathcal{P}^{'}_{u_i,v}$ using Equation 3. This initial part is concluded by setting the lowest profit value (*LowestProfit*) as the *Profit* of the k^{th} customer in R (Lines 3 to 12). Then, the algorithm starts to retrieve customers one by one in the order of their travel penalty score. This can be done using an *incremental* k-nearest-neighbor algorithm. For each customer u_i, we calculate the *maximum possible* profit that i can have by multiplying the travel penalty of u_i $\mathcal{T}_{u_i,v}$ by $P^{'}_{MAX}$, the maximum possible profit value in the system. If u_i cannot make it into the list of top-k customers with this maximum possible profit value, we immediately terminate the algorithm by returning R as the top-k customers without computing the profit (and travel distance) for more customers in the social graph (Lines 17 to 19). The rationale here is that since we are retrieving customers in increasing order of their penalty and calculating the maximum profit that any remaining customer can have, then there is no way that any unprocessed customer can beat the lowest profit value in R. If the early termination case does not arise, the algorithm continues to compute the profit for each customer u_i using Equation 3, insert u_i into R sorted by its profit value (removing the k^{th} customer if necessary), and adjust the lowest profit value accordingly (Lines 20 to 24).

Algorithm 3 Coupon-based Profit Maximization

```
1:  Function Geo_Coupons(v, L, B_v, K)
2:  /* Populate a list R with a set of K customers*/
3:  R ← φ
4:  while R size ≤ K do
5:      u_i ← Retrieve the customer with the next lowest travel penalty
6:      /*Customer Cost must be ≤ Coupon Price */
7:      if C_{u_i,v} ≤ B_v then
8:          Compute P'_{u_i,v} by Equation 3
9:          Insert u_i into R ordered by P'_{u_i,v}
10:     end if
11: end while
12: LowestProfit ← Profit of the k^th object in R
13: /*Get customers one by one in order of their penalty value */
14: while there are more customers to process do
15:     u_i ← Retrieve the next customer in order of penalty score
16:     MaxPossibleProfit ← P'_MAX × T_{u_i,v}
17:     if MaxPossibleProfit ≤ LowestProfit then
18:         return R /* early termination - end query processing */
19:     end if
20:     P''_{u_i,v} ← P'_{u_i,v} × T_{u_i,v} /* Equation 6 */
21:     if P''_{u_i,v} > LowestProfit then
22:         Insert i into R ordered by P''_{u_i,v}
23:         LowestProfit ← Profit of the k^th object in R
24:     end if
25: end while
26: return R
```

To calculate an exact travel penalty for a customer u_i to venue v, we employ an incremental k-nearest-neighbor (KNN) technique [18], [19]. Given a venue v location $\mathcal{L}_v$, incremental KNN algorithms return, on each invocation, the next customer u_i nearest to v with regard to travel distance d. In our case, we normalize distance d to take a value from zero and one to get the travel penalty $\mathcal{T}_{u_i,v}$ in Equation 6. Incremental KNN techniques exist for both Euclidean distance [18] and (road) network distance [19]. The advantage of using Incremental KNN techniques is that they provide an *exact* travel distance

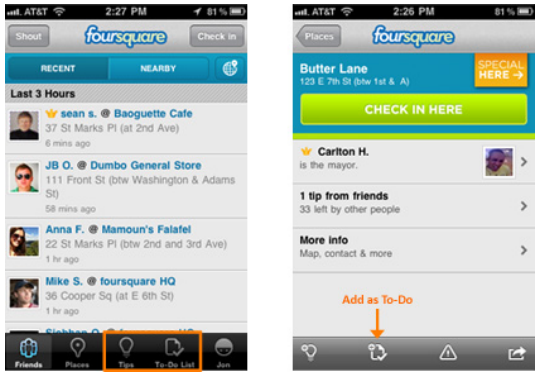


Fig. 4. Foursquare mobile application

value between a venue's location and each customer in the social graph. The runtime complexity of retrieving a single customer using incremental KNN in Euclidean space is [18]: $O(k+logN)$, where N and k are the number of total customers and customers retrieved so far, respectively.

VI. Experimental Evaluation

This section provides experimental evaluation of PLUTUS based on an actual implementation using JAVA 6.

Data Set. All of our experiments are based on a real dataset obtained from Foursquare. Foursquare [20] is a mobile location-based social network application (see figure 4 [4]). We use the publicly available APIs provided by Foursquare to run a crawler that collected results for 4,392 users and 36,963 venues. Users are associated with a home city, and alert friends when visiting a venue (e.g., restaurant) by **"checking-in"** on their mobile phones. During a "check-in", users can also leave **"tips"**, which are free text notes describing what they liked about the venue. Any other user can add the "tip" to her **"to-do list"** if interested in visiting the venue. Once a user visits a venue in the "to-do list" , she marks it as "done". Also, users who check into a venue the most are considered the **"mayor"** of that venue.

Extracting user opinions. Foursquare does not give the functionality to a user to give an explicit rating for a place. To extract user opinions for different venues from the Foursquare data, we map each user visit to a single rating. The *user* and *venue* attributes are represented by the unique Foursquare user and venue identifier, respectively. We use a numeric *rating* value range of [1, 4], translated as follows: (a) 4 represents the user is the "mayor" of the venue, (b) 3 represents that the user left a "tip" at the venue, and (c) 2 represents the user marked a tip left in the venue as "done". (d) 1 represents the user only visited (check-in) the venue.

Building the social graph. Using Foursquare data, we build a social graph of these users and annotate it with influence, cost and profit to allow our algorithm to work. In the generated social graph, each user is represented by a node. Location of each node is assigned to the location of the home city of the associated user. A static cost is assigned to each user according to the total number of friends of this user and her distance to the chosen venue. The cost increases with the user popularity (number of friends). Profit is assigned to a user according to the activity of the user (total number of checkins in the system). The intuition is that a user with higher activity is likely to visit to the designated venue that another user with lower activity. An edge is added to the social graph between two friends or between a brand and a follower. For each pair of friends (u_i, u_l), we add a pair of directed edges, (u_i, u_l) and (u_l, u_i). The influence of a user u_i on his friend u_l is calculated by counting the number of checkins by user u_i to any venue v_j followed by a checkin of u_l to the same venue v_j. We assume here that the u_i visited this place first and tells u_l about this place who was convinced (influenced) to visit the same place later.

In the experiments, we measure quality, processing time and expected profit for the following algorithms:

- *Celeb*: a basic celebrity-based algorithm that selects the most popular users in the social graph that covers the budget.
- *Plutus-Celeb*: represents the celebrity-based profit maximization algorithm presented in section V-A.
- *Coup*: a basic coupon-based algorithm that selects the top-k most popular users in the social graph and assign a coupon to each of them.
- *Plutus-Coup*: represents the coupon-based profit maximization algorithm presented in section V-B (Algorithm 3).

All experiments were performed on an Intel Core2 8400 at 3Ghz with 4GB of RAM machine running Ubuntu Linux 10.04 operating system.

Quality Metric. To measure the quality of our algorithm, we choose a venue and retrieve all checkins on this venue. Then, we find a time point such that 80% of the checkins are before this time (training set) and only 20% are beyond this time (test set). We build the social graph around this venue using the training set. We then pass this social graph to the algorithm and let it choose the best users to attract to the venue. We increment our quality measure by one for each chosen user who actually visited the venue (according to the test set). We further check the friends of each of those users who actually visited the venue, we increment the quality measure by one for each friend who visited the venue after user visit. In other words, we start a breadth first search from the chosen users who visited the venue. For each user in the search, we only traverse his friends who went to the same venue after her visit. The total number of users traversed in this search is the quality measure we use.

A. Plutus-Celeb versus Celeb

In this section, we compare the quality, gained profit, and processing of both the *Celeb* and *Plutus-Celeb* algorithms.

1) Effect of budget on Quality: Figure 5 shows the quality measure for both *Celeb* and *Plutus-Celeb*. We measure the quality while varying the budget to take the values \$5000, \$10000, \$20000 , \$50000, and \$100000. As it turns out from the figure, *Plutus-Celeb* consistently achieves higher quality than *Celeb*. That is explained by the fact that *Plutus-Celeb* takes into account the social ties, the spatial distance, and the user ratings as opposed to *Celeb*. More specifically, the gap between *Plutus-Celeb* and *Celeb* is higher for smaller budget

[4]The screenshot in Figure 4 taken from: http://foursquare.tumblr.com/

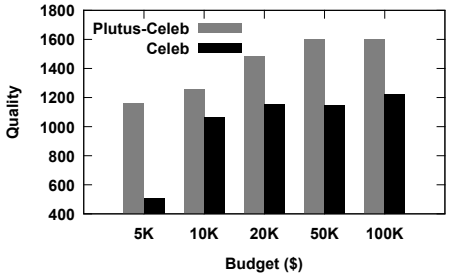


Fig. 5. Effect of budget on quality

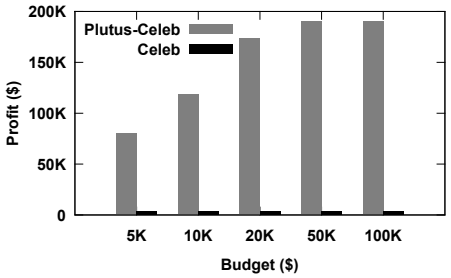


Fig. 6. Effect of budget on profit

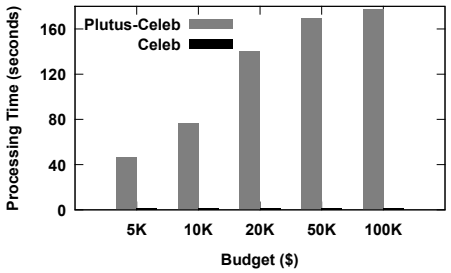


Fig. 7. Effect of budget on time

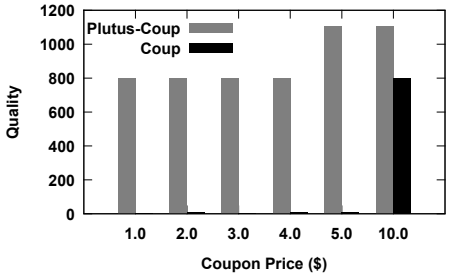


Fig. 8. Effect of coupon price on quality

values. These results manifest that *Plutus-Celeb* is efficient in spending the budget even with tight budget values.

2) Effect of budget on profit: Figure 6 compares the processing time for running both *Celeb* and *Plutus-Celeb*. The processing time is measured in terms of CPU time taken by each algorithm to return the set of recommended customers. We measure the processing for budget values \$5000, \$10000, \$20000 , \$50000, and \$100000. As it turns out from figure 7, *Plutus-Celeb* takes more time to run thatn *celeb* for all budget values. That behavior is explained by the fact that *Celeb* incurs no processing overhead as it uniformly selects customers that match the budget. On the other hand, *Plutus-Celeb* needs to calculate the total profit $\mathcal{P}^{''}_{u_i,v}$ of all users while running the hill climbing algorithm. However, *Plutus-Celeb* takes order of seconds to run which is acceptable especially that the algorithm is expected to be performed offline.

3) Effect of budget on processing time: Figure 7 measures the total expected profit gained using both *Celeb* and *Plutus-Celeb*. We measure the total profit in dollars for the whole budget values \$5000, \$10000, \$20000 , \$50000, and \$100000. As presented in the figure, *Plutus-Celeb* incurs $40\times$ to $180\times$ higher estimated profit than *Celeb* for all budget values. That behavior is explained by the fact that *Celeb* might select users that are not relevant to the designated venue. On the other hand, *Plutus-Celeb* recommends those user that are socially, spatially relevant to the designated venue while taking the users interest in that venue into account.

B. Plutus-Coup versus Coup

In this section, we compare the quality, gained profit, and processing of both the *Coup* and *Plutus-Coup* algorithms.

1) Effect of coupon price / number of coupons on quality: In figure 8, we measure the quality of both the *Plutus-Coup* and *Coup* algorithms. The quality of each algorithm is measured for coupon price values \$1, \$2, \$3, \$4, \$5, and \$10. As shown in Figure 8, increasing the coupon price while keeping total number of coupons fixed, increases total quality of the algorithm for both approaches. The intuition behind this is that as the coupon price increases, both algorithms has a wider choice when selecting people to give coupons and hence there is a high possibility that both algorithms to return high quality answer.

However, *Plutus-Coup* shows consistently higher quality than *Coup* for all coupon price values. This phenomenon is explained by the fact that *Plutus-Coup* efficiently assigns coupons to users based on their social, spatial, and their interest in the designated venue. For coupon price value , the quality of *Coup* jumped closer to *Plutus-Coup* as in this case many users in the system including low cost users accept higher value coupons. That increased the possibility that *Coup* achieves a high quality.

Figure 11 presents the quality of both algorithms while setting the number of coupons ($\mathcal{K}$) to 1, 5, 10, 50, and 100. The figure shows that for small number of coupons *Plutus-Coup* achieves higher quality than *Coup* because *Plutus-Coup* efficiently distributes the small number of coupons on the user with the highest potential. For larger number of coupons, *Coup* quality comes close to *Plutus-Coup*. This happens because with higher number of coupons, the possibility for *Coup* to get high potential users increases.

2) Effect of coupon price / number of coupons on profit: Figure 9 calculates the total estimated profit for both *Plutus-Coup* and *Coup* algorithms. We measure the estimated profit for coupon price values \$1, \$2, \$3, \$4, \$5, and \$10. As it turns out from the figure, *Plutus-Coup* achieves up to $9\times$ higher estimated profit than *Coup*. That happened due to the fact that *Plutus-Coup* picked the most relevant customers to the designated venue.

Figure 12 measures the profit for both algorithms while setting the number of coupons ($\mathcal{K}$) to 1, 5, 10, 50, and 100. As given in the figure, *Plutus-Coup* consistently outperforms

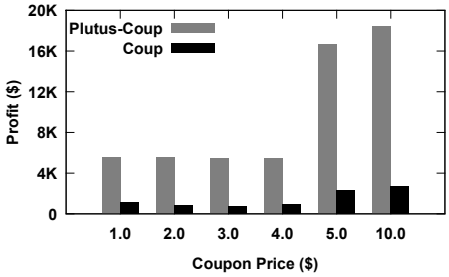


Fig. 9. Effect of coupon price on profit

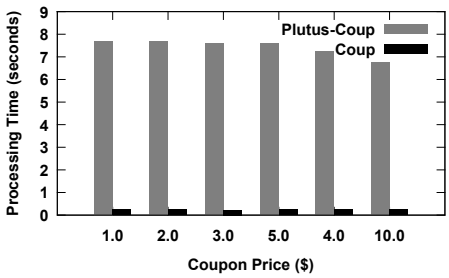


Fig. 10. Effect of coupon price on processing time

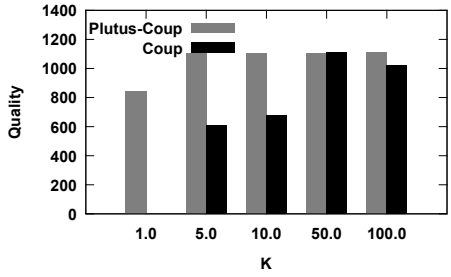


Fig. 11. Effect of number of coupons on quality

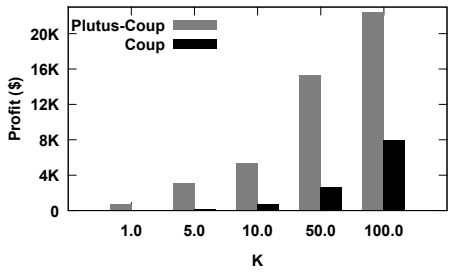


Fig. 12. Effect of number of coupons on profit

Coup for the estimated profit as *Plutus-Coup* is expected to recommend customers with higher value than that of *Coup*.

3) Effect of coupon price / number of coupons on processing time: Figure 10 measures the total processing time of both *Plutus-Coup* and *Coup* algorithms while setting the coupon price values to \$1, \$2, \$3, \$4, \$5, and \$10 . As presented in the figure, *Plutus-Coup* incurs more processing overhead than *Coup*. That is explained by the fact that *Plutus-Coup* needs to calculate the total profit calculation for each distantly close user which leads to more processing time. However, *Plutus-Coup* still terminates only in order of seconds as the algorithm employs an early termination strategy using the travel penalty.

Figure 13 compares both processing time of both *Plutus-Coup* and *Coup* algorithms. As it turns out from the figure, increasing the number of coupons ($\mathcal{K}$) leads an increase in the total processing time of *Plutus-Coup*. This is explained by the fact that *Plutus-Coup* needs to calculate the total profit for more users in the social graph for higher value of $\mathcal{K}$.

VII. Related Work

Informational social influence [2], [21], [22], [23], [24] is the psychological phenomenon that describes the positive influence created when someone finds out that others are doing something, also known as *Social Proof*. Kempe, Kleinberg and Tardos in [2] build the theoretical foundation behind the optimization problem of selecting the most inuential nodes in a social network. Their work aims at maximizing the spread of influence in the social graph. They have shown the problem to be NP-hard, but can be approximated within 63% of the optimal using a a natural greedy search strategy. Zhang et. al studies the geo-social influence of spatial events in location-based social networks [24].

Viral Marketing [3], [4], [25], [26], [27] use social networks as a word of mouth tool that helps achieving marketing objectives through self-replicating viral processes. For instance, a publisher in the phase of new book release, might pre-select a set of *Twitter* users and give them free copies of the book. These Twitter users, if well selected, might substantially help in advertising that new book, by word of mouth, and hence increasing the book sales. Domingos in [25] discusses the importance of viral marketing in non-spatial products (e.g., movies, books) marketing. He argued that using traditional direct marketing may lead to suboptimal marketing decisions. Domingos proposed the idea that online social networks can be leveraged to calculate the network value of customers, and hence making more precise marketing decisions. Richardson and Domingos in [3] views the market as a social network instead of independent customers. They mined a recommender system database to retrieve the social influence among users. The authors in [26], [27] also leverages the social influence to increase profit and maximize a product adoption in online social networks. PLUTUS extends the viral marketing techniques to incorporate both the spatial and users opinions effects.

Location-based Social Networks [28], [29], [30], [31] have been exploited to study the spatial behavior of users and how that behavior is associated with the users social ties [7], [32]. Cho, Myers and Leskovec in [32] have applied extensive data analysis on two data sets from real location-based social networking websites (i.e., Gowalla and BrightKite). They also derived a user mobility model taking into account three sub-models: (a) Model for spatial locations that a user regularly visits, (b) A model for temporal movement between these locations, and (c) A model for movement that is influenced by the social network ties. The authors in [31] proposed a novel method to recommend items to users in location-based social networks. They proposed a taxonomy of location-based ratings that aims at incorporating the geo-location of venues and users in building the recommendation model, resulting into more relevant recommendation results. PLUTUS leverages data generated from existing location-based social networking systems to retrieve the set of users that are expected to maximize the total venue profit.

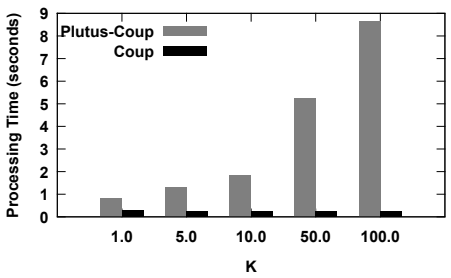


Fig. 13. Effect of number of coupons on processing time

Recommender Systems. A Recommender systems speculates how much a user would like an item she has never seen (bought, watched, ...) before. *Collaborative Filtering* [12] is considered the most popular technique among several recommendation techniques proposed in the literature [12], [9], [33], [34]. There are several methods to perform collaborative filtering including item-item [11], user-user [12], regression-based [11], or approaches that use more sophisticated probabilistic models (e.g., Bayesian Networks [13]). Collaborative filtering techniques analyze past community opinions to find correlations of similar users (or items) to suggest k personalized items (e.g., movies) to a querying user u. Community opinions are usually expressed through explicit ratings represented by the triple (*user*, *item*, *rating*) that represents a *user* providing a numeric *rating* for an *item*. PLUTUS employs the item-based collaborative filtering technique to predict the rating that a user would give to an unseen item.

VIII. Conclusion and Future Work

In this paper, we presented PLUTUS, a marketing framework that aims at recommending customers to grow the business of a particular venue. To this end, PLUTUS leverages social data, spatial data, and user opinions data provided by location-based social networks, and uses them in concert to retrieve those customers that are expected to maximize the total venue profit. We tested the performance of PLUTUS using real data from Frousquare location-based social network. In the future, we plan to deploy our framework in a real life setup to test PLUTUS with real venues (e.g., restaurants).

IX. Acknowledgement

This work is supported in part by the National Science Foundation under Grants IIS-0952977 and IIS-1218168.

References

[1] T. Y. Chan, C. Wu, and Y. Xie, "Measuring the lifetime value of customers acquired from google search advertising," *Marketing Science*, vol. 30, pp. 837–850, Sept. 2011.

[2] D. Kempe, J. M. Kleinberg, and . Tardos, "Maximizing the spread of influence through a social network," in *KDD*, 2003.

[3] M. Richardson and P. Domingos, "Mining Knowledge-Sharing Sites for Viral Marketing," in *KDD*, 2002.

[4] P. Domingos and M. Richardson, "Mining the network value of customers," in *KDD*, 2002.

[5] "About Foursquare: https://foursquare.com/about/."

[6] A. Kittur, E. H. Chi, and B. Suh, "Crowdsourcing User Studies with Mechanical Turk," in *CHI*, 2008.

[7] J. Lindqvist, J. Cranshaw, J. Wiese, J. Hong, and J. Zimmerman, "I'm the Mayor of My House: Examining Why People Use Foursquare - A Social-Driven Location Sharing Application," in *CHI*, 2011.

[8] M. Ye, P. Yin, and W.-C. Lee, "Location Recommendation for Location-based Social Networks," in *GIS*, 2010.

[9] G. Adomavicius and A. Tuzhilin, "Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions," *TKDE*, vol. 17, no. 6, 2005.

[10] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, "Evaluating Collaborative Filtering Recommender Systems," *TOIS*, vol. 22, no. 1, 2004.

[11] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-Based Collaborative Filtering Recommendation Algorithms," in *WWW*, 2001.

[12] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "GroupLens: An Open Architecture for Collaborative Filtering of Netnews," in *CSWC*, 1994.

[13] J. S. Breese, D. Heckerman, and C. Kadie, "Empirical Analysis of Predictive Algorithms for Collaborative Filtering," in *UAI*, 1998.

[14] G. Karypis, "Evaluation of Item-Based Top-N Recommendation Algorithms," in *CIKM*, 2001.

[15] M. J. Carey et al, "On saying "Enough Already!" in SQL," in *SIGMOD*, 1997.

[16] S. Chaudhuri et al, "Evaluating Top-K Selection Queries," in *VLDB*, 1999.

[17] R. Fagin, A. Lotem, and M. Naor, "Optimal Aggregation Algorithms for Middleware," in *PODS*, 2001.

[18] G. R. Hjaltason and H. Samet, "Distance Browsing in Spatial Databases," *TODS*, vol. 24, no. 2, pp. 265–318, 1999.

[19] K. Mouratidis, M. L. Yiu, D. Papadias, and N. Mamoulis, "Continuous nearest neighbor monitoring in road networks," in *VLDB*, 2006.

[20] "Foursquare: http://foursquare.com."

[21] M. Deutsch and H. B. Gerard, "A study of normative and informational social influences upon individual judgment.," *The Journal of Abnormal and Social Psychology*, vol. 51, no. 3, pp. 629–636, 1955.

[22] H. C. Kelman, "Compliance, identification, and internalization: Three processes of attitude change," *Journal of Conflict Resolution*, vol. 2, no. 1, pp. 51–60, 1958.

[23] R. B. Cialdini, *Influence: Science and Practice*, vol. 3rd. Allyn and Bacon, 2001.

[24] C. Zhang, L. Shou, K. Chen, G. Chen, and Y. Bei, "Evaluating geo-social influence in location-based social networks," in *CIKM*, 2012.

[25] P. Domingos, "Mining social networks for viral marketing," *IEEE Intelligent Systems*, vol. 20, no. 1, pp. 80–82, 2005.

[26] W. Lu and L. V. S. Lakshmanan, "Profit maximization over social networks," in *ICDM*, 2012.

[27] S. Bhagat, A. Goyal, and L. V. S. Lakshmanan, "Maximizing product adoption in social networks," in *WSDM*, 2012.

[28] M. Sarwat, J. Bao, A. Eldawy, J. J. Levandoski, A. Magdy, and M. F. Mokbel, "Sindbad: A Location-based Social Networking System," in *SIGMOD*, 2012.

[29] M. Sarwat, J. Bao, A. Eldawy, J. J. Levandoski, A. Magdy, and M. F. Mokbel, "The Anatomy of Sindbad: a Location-Aware Social Networking System," in *Proceedings of the 5th International Workshop on Location-Based Social Networks*, 2012.

[30] J. J. Levandoski, M. Sarwat, A. Eldawy, and M. F. Mokbel, "LARS: A Location-Aware Recommender System," in *ICDE*, 2012.

[31] M. Sarwat, J. J. Levandoski, A. Eldawy, and M. F. Mokbel, "LARS*: A Scalable and Efficient Location-Aware Recommender System," in *TKDE*, 2013.

[32] E. Cho, S. A. Myers, and J. Leskovec, "Friendship and mobility: user movement in location-based social networks," in *KDD*, 2011.

[33] M. D. Ekstrand, M. Ludwig, J. A. Konstan, and J. T. Riedl, "Rethinking the recommender research ecosystem: reproducibility, openness, and lenskit," in *RecSys*, 2011.

[34] J. J. Levandoski, M. Sarwat, M. D. Ekstrand, and M. F. Mokbel, "RecStore: An Extensible and Adaptive Framework for Online Recommender Queries inside the Database Engine," in *EDBT*, 2012.