International Journal of Advance Research in Science and Engineering
Vol. No.5, Issue No. 04, April 2016
www.ijarse.com
IJARSE
ISSN 2319 - 8354

MIRS: A MAPREDUCE-BASED ITINERARY RECOMMENDATION SYSTEM IN LOCATION-BASED SOCIAL NETWORKS

Madhu K. P.[1], D. Manjula[2]

[1, 2] *Department of Computer Science and Engineering*

Anna University, Chennai (India)

ABSTRACT

With the growth of internet and the information system domain, the number of people who use internet services like social networking, shopping, banking and recommendations have turned to be unimaginable. One of the pertinent research areas in travel recommendation domain is the travel itinerary recommendation. This leads to the need for an efficient and effective travel itinerary recommendation system. In this paper, we propose such a system based on a hybrid method. The system is built based on a parallel design making use of the mapreduce paradigm. An exhaustive analysis on the evaluation of the system shows that the system is proficient enough to provide a more appropriate travel itinerary recommendation to the users with much less response time.

Keywords: Community, Location Based Social Networks, Point of Interests, Location Recommendation, Mapreduce, Personalized Travel Itinerary Recommendation

1. INTRODUCTION

With the increased adoption of gps enabled devices and gps based services; there is an exponential growth in the number of people using the same for various purposes in their day to day life. Recommendation services constitute one such field coined up greatly in the context. A scalable location and itinerary recommendation system is proposed in this article. The locations are determined based on the location history of the user as well as his fellow members in the community. Cold start problem for users is handled by the proposed hybrid recommendation system. An apriori based technique is deployed for the same. If there are not enough candidate locations for generating the itinerary for the user, sufficient number of most similar locations are added to the candidate locations determined, already. The main objective of the Travel Itinerary Recommendation System is to assist the customers to find out venues or locations to visit when he/she is planning for a trip in a city or a region. The system can also be used when the user is new to the city or the region and have enough time to go for such a trip without much of the pre planning.

Consider a special case of the problem where the user has checked-in to a location currently and he wishes to visit some locations in the region where his current location belongs to. It is also assumed that the person has stringent temporal and spatial constraints. He has to start from his current location and has to return back to a specified destination. The trip should be completed within a specified stipulated time after visiting maximum locations in the region. Now this has turned to be an optimization problem. An example scenario is discussed here. A person *P* has checked-in to ITC Chola in Chennai city. He has actually come to Chennai to attend a conference held in ITC Chola. By 5 pm on the same day all his appointments were successfully over and his return flight to his native is by 8 pm the next day. He is totally free from 5 pm on that day to 8 pm the next day. So, he can plan for a trip in the city covering maximum number of venues in the city matching his preferences and interests. This user can use the proposed system to get an optimized itinerary planned for his trip. The system will ask the person to input his current location, timing constraints, destination location etc. The system will then find out the location history of the user and that of the similar users in similar contexts. The system will then determine the optimized itinerary for the user and recommend it to the user. The user can fix the trip according to the itinerary recommended. The user can go for the trip and complete the trip within the stipulated time. The itinerary suggested will contain the locations matching his preference and it will be optimized in such a manner that it will cover maximum locations in the region with minimum time and minimum cost and with maximum possible time availed in each location as per each location's significance with respect to the user preference.

While modeling traveler's itinerary and time, this problem needs to take into account certain subjective or context dependent factors like traveler's preference on venues or locations and his/her emotion. This makes it impossible to model the system with an objective function. So, this problem can be regarded as a tacit multi objective decision making problem. In these kinds of problems, the target cannot be quantified and will be subjective and dependent on the context and the user involved. For instance, the user's mood or preferences may vary dynamically based on which the suggestions should also differ. That is why we refer to this problem as a personalized travel itinerary recommendation system.

LBSN is a network consisting of individuals and their interconnections defined in terms of their relations. The peculiarity of LBSN is that they generate and propagate location-tagged data. This data could be extracted and used for deducing the behavior of users, predict items, locations or even travel itinerary to the users. The location-tagged data along with other prominent attributes could well model the user behavior in the system. The other attributes that are taken into account are viz. time, distance, and rank of the location with respect to the user. The rank of the location is determined primarily by using the tips left by the users who have already visited the location. The tips may have positive or negative polarity. The polarity will influence positively or negatively the other user who is trying to get a suggestion for making an itinerary over the same or nearby similar locations. Some of the prominent LBSNs are viz. Facebook Places, Google Plus, Google Latitude, Gowalla, Brightkite, Twitter, Flickr, Foursquare etc. They share information in the form of text, image, audio or video. Amidst the dangers of privacy concerns, these location-tagged data are widely used in various applications. These sites are able to share the check-in, check-out details of users in specific locations and their tips or responses in these checked-in locations to others who would use this data. Some of the sites provide necessary APIs for users to access the required data and while for some other sites, stored data are available for download. In either case the aspirants are able to access and manipulate the data for further use. They could analyze the data and determine the check-in patterns of users or the personal preferences of the users. These preferences are a better means towards finding and recommending locations/ itineraries to users or similar users in the future. There are other attributes like time, cost, distance, connectivity, season, weather conditions etc. which may be considered along with the check-in pattern to better recommend the locations/ itineraries to the users. Here, the authors assume that the places already visited by the users need not be recommended to them and it is better to recommend locations/ itineraries which are not visited by them but by the members of the community to which the user belongs. The locations already visited by users are known to them and studies reveal that at least 60% of the people prefer to visit new locations rather than already visited locations.

The authors have used data from Foursquare, Brightkite and Gowalla etc. for various experiments and studies. Foursquare allows users to check-in at specific locations and publish their check-in details to Twitter/ Facebook. The aspirants could download the necessary data from either Twitter/ Facebook or from Foursquare directly satisfying necessary authentications [1].

The organization of topics in this article is as follows: the next section discusses about the related works. Followed by this is the proposed methodology. Subsequent to that are the experimental results, the conclusion, acknowledgement and the references.

II RELATED WORKS

Studies show that users with similar location history are more likely to have similar interests and preferences [2] [3]. Eagle and Pentland justifies in their articles that the user's historical behavior is a strong indicator of the user's preferences [4] [5]. This suggestion is more strongly supported by the work by [6] which finds out that a user's historical behavior gathered in an LBSN is more effective and accurate than his online behavior in giving a user's preferences, patterns, interests and experience. Location recommendation system using Collaborative Filtering (CF) model is proposed by [7]. These systems give personalized recommendations for locations by taking into consideration other users' ratings. Each work gives a different version or enhancement to the basic CF method. But most of these systems fail miserably in handling the cold start problem. [8] presents a location recommendation system that incorporates the user's preferences, the user's social connections and the geographic distance between the user and the candidate locations. But, these existing systems are not taking into account the comments given by the visitors on these locations which play a vital role in influencing their friends. These systems suffer from problems viz. cold start problem and sparse rating.

A greedy based strategy for community detection is discussed in [9]. This algorithm consumes more time for execution and it does not consider the user preferences. Conflict graph based community detection is explained in [10]. This work also does not take into account the user preferences. It also consumes more execution time. [11] discusses about TOP and Naïve Bayes algorithms for ranking the locations/items for recommendation.

[12] discusses a work on automatic generation of itineraries from POI graph created from photo streams taken by users. But it gives only an approximate solution. Its execution time is also more. It also requires more user intervention. [13] proposes an automated itinerary planning system for holiday travel. This is purely a commercial model which requires more user intervention. It also consumes more execution time. [14] proposes a travel package recommendation system using COPE. This system deploys genetic algorithm which requires more execution time. [15] proposes an interactive itinerary planning system. It requires more user intervention. It

refines the recommendation in each step by considering feedback from the users in each step. User reluctance in giving the feedback is an issue. [16] has proposed a customized travel planner using mapreduce and approximation algorithms. The system generates single day itineraries in one stage and combines them to give multiday itineraries in the second stage. But it does not incorporate mapreduce in all stages. It also does not consider the tips or the responses given by users for determining the locations in the itinerary. It also requires more user intervention. [17] proposes an automatic travel itinerary planning system for domestic areas of Taiwan. It uses greedy approach for location identification. But it consumes more execution time. [18] proposes a travel package recommendation system using near POI and ranking technique. But its precision is low.

III THE PROPOSED ITINERARY RECOMMENDATION SYSTEM

It is a hybrid recommendation system which integrates the essence of Collaborative Filtering based and content based systems. The tips about different locations given by the users are considered to enhance the recommendations. The user location history and the tips left by the users at these locations are the main inputs for the system. This is a stand-alone itinerary recommendation system. This method gives a recommendation which depends on the individuals' behavior and location history. Also, it contributes to the formation of personalized or customized aggregations in which the diversity in the source or the input tastes will enhance the recommendation.

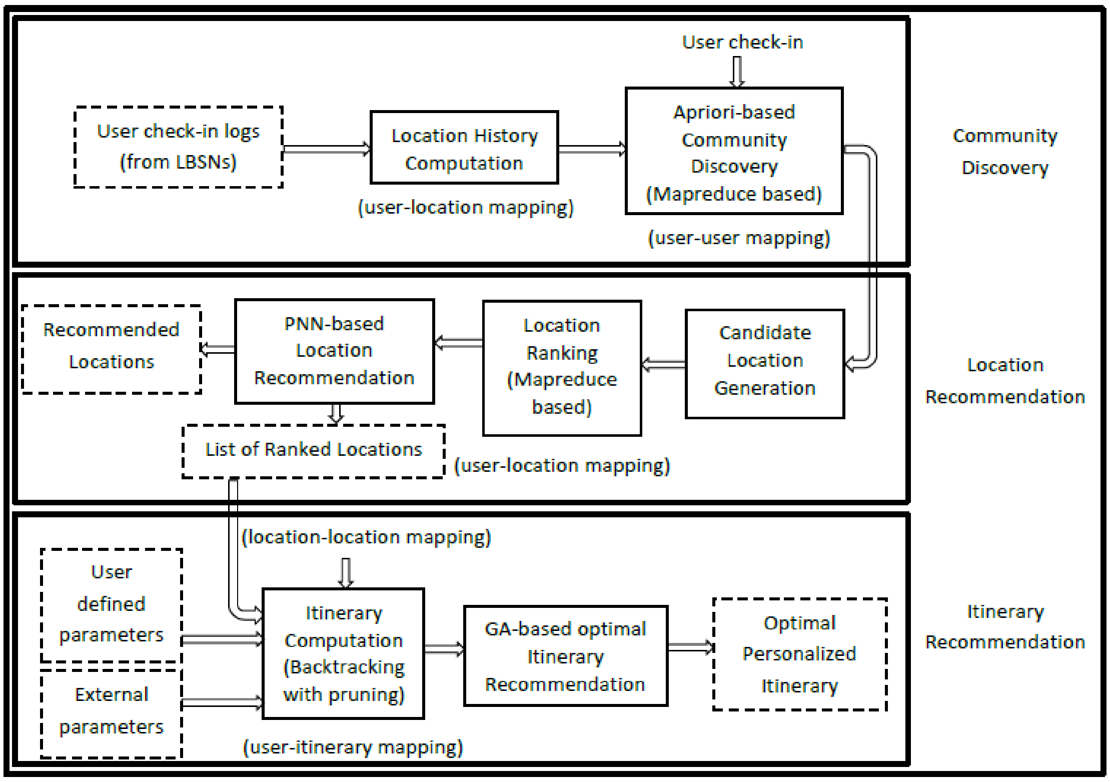
User check-in
User check-in logs (from LBSNs)
Location History Computation
Apriori-based Community Discovery (Mapreduce based)
(user-location mapping)
(user-user mapping)
Community Discovery
Recommended Locations
PNN-based Location Recommendation
Location Ranking (Mapreduce based)
Candidate Location Generation
List of Ranked Locations
(user-location mapping)
Location Recommendation
(location-location mapping)
User defined parameters
External parameters
Itinerary Computation (Backtracking with pruning)
GA-based optimal Itinerary Recommendation
Optimal Personalized Itinerary
(user-itinerary mapping)
Itinerary Recommendation

Figure 1: Itinerary Recommender System Architecture

The architecture of the proposed itinerary recommendation system is given in fig. 1. It has three different segments viz. community discovery, location recommendation and the itinerary recommendation. An overview of each the three segments are given in the following subsections.

3.1 Community Discovery

This segment is responsible for determining the community to which the user belongs. The community discovery is significant in the sense that the recommendation is based on the responses given by the fellow members of the community to the locations already visited by them.

3.1.1 Input data generation

The input data is the user check-in log extracted from the location based social networks. The authors have asked some scholars whom they have acquaintance with, to create account in foursquare, a prominent location based social network. Then, these users were asked to share their check-ins in Twitter. The TwitterAPI was used to extract the check-in data shared by the corresponding users. This data was used for the experimental analysis. Also, the Stanford University Research Centre has published quite a lot of dataset spanning various domains for free download. They have published dataset from Gowalla and Brightkite which are famous location based social networks. The authors have gracefully used these datasets downloaded from the mentioned sites for their experiments.

3.1.2 Location History Computation

The input dataset obtained consists primarily of user check-ins at different locations. It includes the user id, location id, check-in and check-out time and the comments/tips left by them at these locations. These are in fact the individual check-ins and not usually part of the itinerary, but, some of them may be. So, these check-in logs are analyzed and location history pertaining to the users is created. Location history is the sequence of locations visited by the users with their details.

3.1.3 Apriori-based Community Discovery

Community Discovery is one of the pertinent functions in the first segment of the recommendation system. This is implemented by means of mapreduce paradigm to make the system more effective and efficient. The user's community is being determined from the location history of the users. An enhanced apriori based algorithm is deployed for this. This is with the notion that, there is more chance that similar users will visit same or similar locations than others. Based on this fact, the users are grouped into different clusters in a hierarchical manner. Initially, the frequent locationsets are formed using the enhanced mapreduce based apriori algorithm. Then the users are identified to be in different groups or communities based on the presence of these locationsets in their location history. So, one user may be in different communities simultaneously. But, during candidate location generation, the community with which he/ she is associated by means of the most frequent locationset is taken first and then the others till the least frequent locationset.

3.2 Location Recommendation

The second segment of the itinerary recommendation system is the location recommendation segment. This segment is meant to identify and recommend the locations from which the itinerary may be generated and recommended to the users.

3.2.1 Candidate Location Generation

Candidate locations are the locations visited by at least one of the fellow members of the community to which the user belongs. The peculiarity of these locations is that, these locations should not be visited by the users for whom the recommendation is to be made. This is with the assumption that the locations already visited by the user will be known to the user than other locations and these need not be recommended to the user again. Also, studies reveal that more than 65% of the users prefer to go to new locations than the already visited locations. This task is also implemented using a mapreduce algorithm. This makes the task execute faster than the conventional algorithm.

3.2.2 Location Ranking

Location ranking is a laborious task in this segment. So, this task is also implemented using a mapreduce algorithm. While realizing this task, the tips left by the users in these candidate locations are analyzed. Then, their sentiment score is calculated along with its decay. The decay of a tip refers to the loss in relevance of the tip over time. The sentiment score is multiplied by the decay and they are summed up for each of the locations. Then this value is normalized over a stipulated period. This aggregated weight is regarded as the rank of the location. This parallel algorithm contributes to the reduction in execution time of the system.

3.2.3 Probabilistic Neural Network-based Location Recommendation

The next phase consists of a probabilistic neural network based recommender system. We have converted the recommendation problem as a classification problem with the regarded as output classes. These locations are labeled categorically with their location names. Probabilistic neural networks are faster in training and their accuracy is in classification is far better than its counterparts. Its architecture is given in fig. 2. PNN consists of four layers of neurons. The input layer accepts input given in the form of a feature vector which is a 6-tuple, F = {userid, sim_userid, day_of_week, time_slot, rank, locationid}. Next layer is the hidden layer which takes input from the input layer and the learning is performed in this layer. The output from this layer is passed to the summation layer which is responsible for inducing the probability of the features for being in different classes/ locations. The output layer classifies the particular input set into the correct class/ location. The classified location is recommended to the user. The user who anticipates only the individual location recommendation may carry on with this recommendation. The system can be tuned to give recommendations consisting of one or more locations depending on the context. This is a subsidiary version of the eventual system.

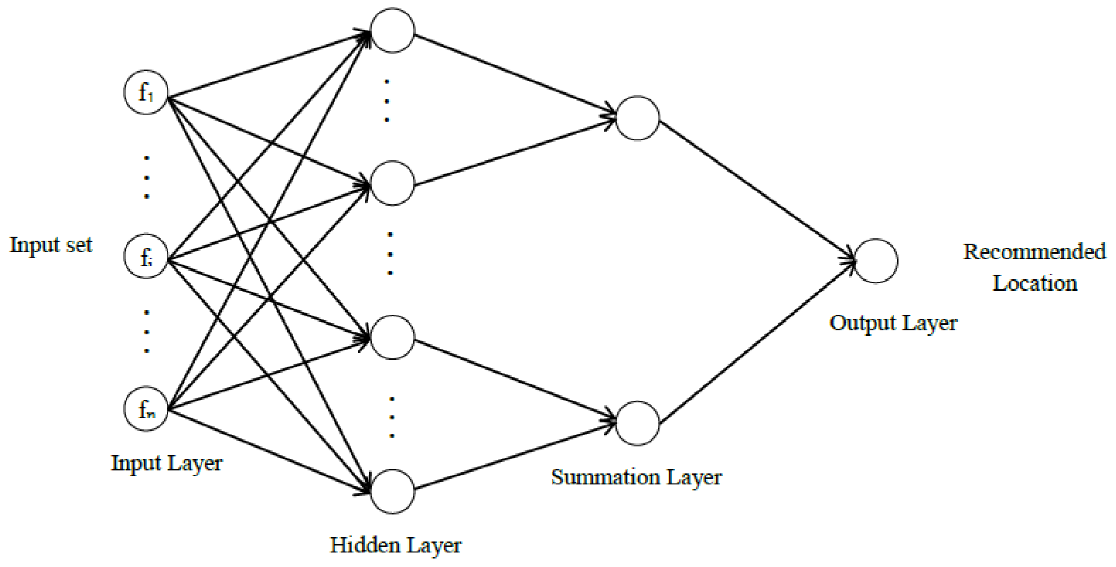

Figure 2: Architecture of Probabilistic Neural Network

3.3 Itinerary Recommendation

The last segment of the itinerary recommendation system is the itinerary recommendation segment. The itinerary is generated out of the location recommended to the user. Apart from this input, this segment takes external inputs for improved itinerary generation and recommendation.

3.3.1 Itinerary Computation using backtracking and pruning algorithm

This is the most prominent task in the entire system. It accepts inputs like the starting location, the destination location and the time available for the itinerary, from the user. It also accepts external inputs like the weather conditions of the location and the visit timing of the particular location and whether the location is indoor or outdoor. The initial itinerary consists only of the starting location. Then the system picks each of the locations and checks its feasibility of inclusion in the itinerary. While including a location in the itinerary, the location type is checked to be either indoor or outdoor as weather conditions usually affects the outdoor locations. The visit timing at the location is also a factor to be considered. Some locations/venues will have stringent timing requirements to be satisfied for the visit while some others may have desirable timing for visits. The system takes into account all these factors. If the location selected satisfies all these criteria then the location is added to the current itinerary. The time for the whole itinerary is updated. The time taken to reach the location from the current path and the average time spent by the visitors in this location are added to the time of the current itinerary. The cost of the itinerary is also updated in the same way by adding the cost to reach the location from the current itinerary and the cost incurred in the location during the visit. The system then picks the next location and does the same operations. The total time of the itinerary should not exceed the time available for the itinerary. If some itinerary violates this condition then that itinerary is dropped and the itinerary computation proceeds by checking with the next location. Meanwhile, if an itinerary reaches the destination, then the itinerary is saved to the itinerary list. The destination is removed from the itinerary. This is called as pruning the itinerary. The system then continues to find the next itinerary with the other locations. This is called as backtracking. The system is constrained by the fact that the itinerary recommended should contain optimum number of locations and should provide at least a time which is not less than the average time spent by the users who have already visited the location. The set of itineraries is then given to the next phase for determining the optimal itinerary for recommendation.

3.3.2 Genetic Algorithm-based Optimal Itinerary Recommendation

Genetic Algorithm is a technique used for finding the optimum itinerary from the partial solutions which are the itineraries in the itinerary list. Genetic Algorithm operates on the principle of natural evolution. The operands are called as chromosomes. The collection of chromosomes constitutes the population. The chromosomes represent the partial itineraries. A fitness function is associated with the algorithm. The fitness function defines the quality of the itinerary. The itineraries are ranked and the two best itineraries are selected for crossover operation. A new itinerary is formed as a result of the crossover operation. Its fitness will be better than that of the parent itineraries. The weak itineraries are removed from the itinerary set. This process is continued until the optimal itinerary is found out. The fitness function for the application is defined as below.

$$\underset{\max}{f(I)} \leftarrow ((w_1 \times I.num\_locs + w_2 \times I.time + w_3 \times (I.staytime - I.traveltime)) - w_4 \times I.cost) \quad (1)$$

The optimal itinerary is then recommended to the user. Since this is a personalized itinerary recommendation system, the recommendation will vary for each of the users.

IV THE MAJOR ALGORITHMS IN THE SYSTEM

The key algorithms instigated in the system are discussed in detail in this section.

Algorithm1: ***Location_Recommendation* ( )**

Input: (1) *check-in_log* (2) user u_r

Output: A List of recommended locations for user u_r

Steps

```
1.  Start
2.  Access the check-in log crawled from LBSNs
3.  $LH \leftarrow \Phi$ for all users  //Location History
4.  for each user $u_i$
a.  Compute_LH( check-in_log)
b.  end for
5.  $Compute\_locationset()$
6.  $Community\_Discovery()$
7.  if $c = \Phi$ for $u_i$ then
a.  $c \leftarrow Find\_Community(u_r)$
```

```
b.      $c \leftarrow c \cup \{u_r\}$
c.      **end if**
8.      **for** each cluster c in C **do** // C is the set of all communities
a.      $profile_c \leftarrow Compute\_profile(c)$
b.      $REP_c \leftarrow Compute\_rep\_User(profile_c)$
c.      $u_c \leftarrow Compute\_Community(u_r)$
d.      $L_c \leftarrow Compute\_candidate\_locations(u_c)$
e.      $L_R \leftarrow Compute\_LocationRank(L_c)$
f.      **end for**
9.      Pass $L_R$ to Itinerary Recommendation phase for itinerary computation
10.     Submit the ranked locations and associated data to PNN for finding the location to be recommended
11.     Recommend the location given by PNN as the output //Output for location recommendation
12.     end
```

Algorithm2.1: *Compute_locationset* ( )

Map Task: // One for each split

Input: S_i // Split i

Output: <*key*, 1> pairs where key is an element of the candidate locationset

Steps

```
1.      start
2.      map( IntWritable key, Text value, Context context) // Map function
a.      String []inputTokens ← value.toString().split("\t")
b.      String userid ← inputTokens[0]
c.      String locationid ← inputTokens[1]
d.      for each element l in L // L ← powerset of all locationid
    i.      context.write( l, 1)
    ii.     end for
e.      end map
3.      end
```

Reduce Task:

Input: <*key2*, *value2*> pairs, where key2 is an element of the locationset and value2 is its occurrence in each split

Output: <*key3*, *value3*> pairs, where key3 is an element of the locationset and value3 is its occurrence in the whole dataset

Steps

```
1.      start
2.      reduce(IntWritable key2, Iterable<Text>  value2) //Reduce function
a.      count ← 0
```

```
   b.    while ( value2.hasNext( )) do
  i.        count← count+ value2.getNext( )
 ii.     end while
   c.    context.write( key2, count)
   d.    end reduce
3.    end
```

Algorithm2.2: ***Community_Discovery* ( )**

Map Task // One for each split

Input: <*key3*, *value3*> pairs, where key3 is an element of the locationset and value3 is its occurrence in the whole dataset, check-in_log

Output: <*key3*, *userid*> pairs, where *key3* is an element of the candidate locationset and *userid*

Steps

```
1.    start
2.    map( IntWritable key3, Text value, Context context) // Map function
   a.    for each line key3 in $LH_i$
  i.        context.write( key3, userid)
 ii.     end for
   b.    end map
3.    end
```

Reduce Task

Input: <*key3*, *value3*> pairs, where *key3* is an element of the locationset and value2 is *userid* if *key3* is in *LH*

Output: <*key3*, *value3*> pairs, where key3 is an element of the locationset and value3 is its occurrence in the whole dataset

Steps

```
1.    start
2.    reduce(IntWritable key3, Iterable<Text>  value3) //Reduce function
   a.    $C \leftarrow \Phi$
   b.    while ( value3.hasNext( )) do
  i.        $u \leftarrow$ value3.getNext( )
 ii.        $C \leftarrow C \cup \{u\}$
iii.     end while
   c.    context.write( key3, C)
   d.    end reduce
3.    end
```

Algorithm3: ***Compute_candidate_locations* ( )**

Map Task: // One for each split

Input: (1) S_i // Split i (2) u_c - the community for u_i , (3) user u_i

Output: <*key*, 1> pairs where key is an element of the candidate locationset

Steps

```
1.   Start
2.   map( IntWritable key, Iterable <Text> value, Context context) // Map function
a.   String []inputTokens ← value.toString().split(“\t”)
b.   String userid ← inputTokens[0]
c.   String locationid ← inputTokens[1]
d.   for each $u' \in u_c$
i.   if $u_i$ has not visited the location with id as locationid and $u'$ has visited
1.   context.write( locationid, 1)
2.   end if
ii.  end for
e.   end map
3.   end
```

Reduce Task:

Input: <*key2*, *value2*> pairs, where key2 is a location and value2 is its occurrence

Output: <*key3*, *value3*> pairs, where key3 is a location and value3 is its occurrence in the *LH* of the community members

Steps

```
1.   start
2.   reduce(IntWritable key2, Iterable<Text>  value2) //Reduce function
a.   count ← 0
b.   while ( value2.hasNext( )) do
i.   count← count+ value2.getNext( )
ii.  end while
c.   context.write( key2, count)
d.   end reduce
3.   end
```

Algorithm4: *Compute_LocationRank* ( )

Map Task: // One for each split

Input: (1) S_i // Split i (2) u_c - the community for u_i , (3) user u_i

Output: <*key*, 1> pairs where key is an element of the candidate locationset

Steps

```
1.   Start
2.   map( IntWritable key, Text value, Context context) // Map function
```

```
a.      String []inputTokens ← value.toString().split("\t")
b.      String userid ← inputTokens[0]
c.      String locationid ← inputTokens[1]
d.      for each $L' \in L$ do
i.      for each user $u' \in u_c$ do
ii.     $L'.tip_{u'} \leftarrow Extract\_Tip(Tip, L', u')$ //tip uploaded by $u'$ on $L'$
iii.    $L'.ts_{u'}$ ← timestamp of the visit/tip by $u'$ on $L'$
iv.     $SS_{u'} \leftarrow Compute\_SentiScore(L'.tip_{u'})$
v.      $decay_{u'} \leftarrow 1-((L'.ts_{u'} - \mathrm{t})/T)$
        //decay is the loss in relevance of the tip
        // t is the current time
        // T is the time range – 5 years
vi.     end for
vii.    end for
a.      $pL'.rank_{u_i} \leftarrow (\sum decay_{u'} \times SS_{u'})$
b.      context.write( $L'$, $pL'.rank_{u_i}$ )
c.      end map
3.      end
```

Reduce Task:

Input: <*key2*, *value2*> pairs, where key2 is an element of the locationset and value2 is its occurrence in each split

Output: <*key3*, *value3*> pairs, where key3 is an element of the locationset and value3 is its occurrence in the whole dataset

Steps

```
1.      start
2.      reduce(IntWritable key2, Iterable<Text>  value2) //Reduce function
a.      rank ← 0
b.      count ← 0
c.      while ( value2.hasNext( )) do
i.      rank ← rank+ value2.getNext( )
ii.     count ← rank+1
iii.    end while
d.      $L'.rank_{u_i} \leftarrow rank / count$
e.      context.write( $L', L'.rank_{u_i}$ )
f.      end reduce
3.      end
```

Algorithm5: ***Itinerary_Computation* ( )**

//Backtracking with pruning

Input: (1) A connected graph consisting of recommended locations (2) s - the start location, (3) d – the destination location, (4) time_available, time duration of the itinerary

Output: A set of itineraries

Steps

```
1.    Start
2.    Path $Itineraries[] \leftarrow \Phi$ ; $I \leftarrow$ s; $P.time \leftarrow 0$; $P.cost \leftarrow 0$
3.    NS.push(s) //push to the stack, NS
4.    while ( !isEmpty(NS)) do
a.    $L' \leftarrow$ NS.pop( )
b.    nb[]← find_neighbours( $L'$ )
c.    NS.push(nb) // Push all the elements to the stack
d.    if isConnected( $I$ , $L'$ ) and ! $I$ .contains( $L'$ )
i.    if $L'$ == d
1.    $I \leftarrow I.append(L')$
2.    $I.time \leftarrow I.time + getTime(I, L')$
3.    $I.cost \leftarrow I.cost + getCost(I, L')$
4.    Write ( $Itineraries$ , $I$ , $I.time$ , $I.cost$ )
5.    $I.prune(L')$ //Remove $L'$ from the itinerary
6.    continue
ii.   if (( $L'.type$ == indoor ) or ( $L'.type$ ==outdoor && $L'.weather$ == OK && $L'.timing$ == OK))
1.    $time \leftarrow I.time + getTime(I, L') + getTime(L')$
2.    if $time \le time\_available$
3.    $I \leftarrow I.append(L')$
4.    $I.time \leftarrow I.time + getTime(I, L') + getTime(L')$
5.    $I.cost \leftarrow I.cost + getCost(I, L') + getCost(L')$
6.     end if
7.     end if
iii.  end if
e.    end while
5.    WritetoFile( $Itineraries$ )
6.    end
```

Algorithm6: ***GA_Optimal_Itinerary_Recommendation* ( )**

//Genetic Algorithm-based Optimal Itinerary Recommendation

Input: (1) A set of itineraries

Output: An optimal itinerary for recommendation

Steps

```
1.      Start
2.      Chromosomes are initialized with the itineraries
3.      Compute the fitness function for each of the itineraries
        $f_{\max}(I) \leftarrow ((w_1 \times I.num\_locs + w_2 \times I.time + w_3 \times (I.staytime - I.traveltime)) - w_4 \times I.cost)$
4.      Rank the itineraries according to the value of the fitness function
5.      Select best two itineraries as parents based on the rank
6.      Do crossover with probability $C_{prob}$
7.      Do mutation with probability $M_{prob}$
8.      Retain the new itineraries in the set of itineraries
9.      Remove the itinerary with the least fitness value
10.     **if** termination condition is not accomplished then repeat the steps 3 to 10
**else**
a.      Find the best ranked itinerary and recommend to the user
**end if**
11.     end
```

Algorithm 1 gives the steps for accepting the input dataset and generating the location history for each of the users. It then invokes the community detection algorithm. Afterwards, the algorithm invokes the candidate location generation and location ranking algorithms. Algorithm 2.2 lists the steps for community discovery. As part of community discovery it invokes Algorithm2.1 which generates the frequent locationset. This algorithm is an enhanced version of the apriori algorithm for frequent itemset mining. Algorithm 3 is responsible for candidate location generation. Algorithm 4 does the location ranking. The ranked locations along with other details are fed to the Probabilistic Neural Network for recommendation. The PNN-based recommender system is a classifier which takes the input set with the features like the userid, similar user's id, day and time of the check-in proposed, locationid, venueid etc. It then classifies the input data set to the correct location. Algorithm 5 is used for itinerary generation. The algorithm is a backtracking algorithm with pruning. It starts with the start location as the only location in the itinerary. Then it checks whether each of the locations can be added in the itinerary or not. The location added should satisfy certain critical criteria. After finding all the itineraries, they are passed to Algorithm 6 which is Genetic Algorithm for finding the optimal itinerary. The optimal itinerary is the one which satisfies equ. (1). This itinerary is recommended to the user. All the algorithms except algorithm 5 are mapreduce based algorithms. They will be executed in parallel, thereby reducing the execution time.

IV EXPERIMENTAL EVALUATION AND RESULTS

As part of the experimentation and performance analysis, the mapreduce algorithms were executed on a 4-node Hadoop cluster. The cluster was setup with all the four nodes of same hardware and software configuration. All thenodes were of Intel core i7 2.3 GHz processor with 8 GB RAM. The OS is Ubuntu 15.04 and the Hadoop version is Hadoop1.2.1. The dataset was collected from Foursquare and also from Twitter. Another set was downloaded from Stanford University site. The experimental analysis is the same procedure followed in [1].

The proposed recommendation system using hybrid technique is experimentally compared with the systems which are based on collaboration filtering technique and content based techniques. Significant improvement in performance is obtained. As part of a comprehensive analysis, particular nodes with known activity history, interests and profiles were selected and the system was made to recommend locations for the known user. Our system is found to give more accurate recommendations compared to its counterparts. The users were asked to check-in at a particular location and post a query for the recommendation process to get initiated. The response times were analyzed and studied in detail.

Figure 3 - 7, shows the comparison of execution time between the proposed and the existing algorithms implemented in the personalized itinerary recommendation system. It is obvious that the proposed algorithm for each of the modules in the system clearly outperforms the other algorithms being compared.

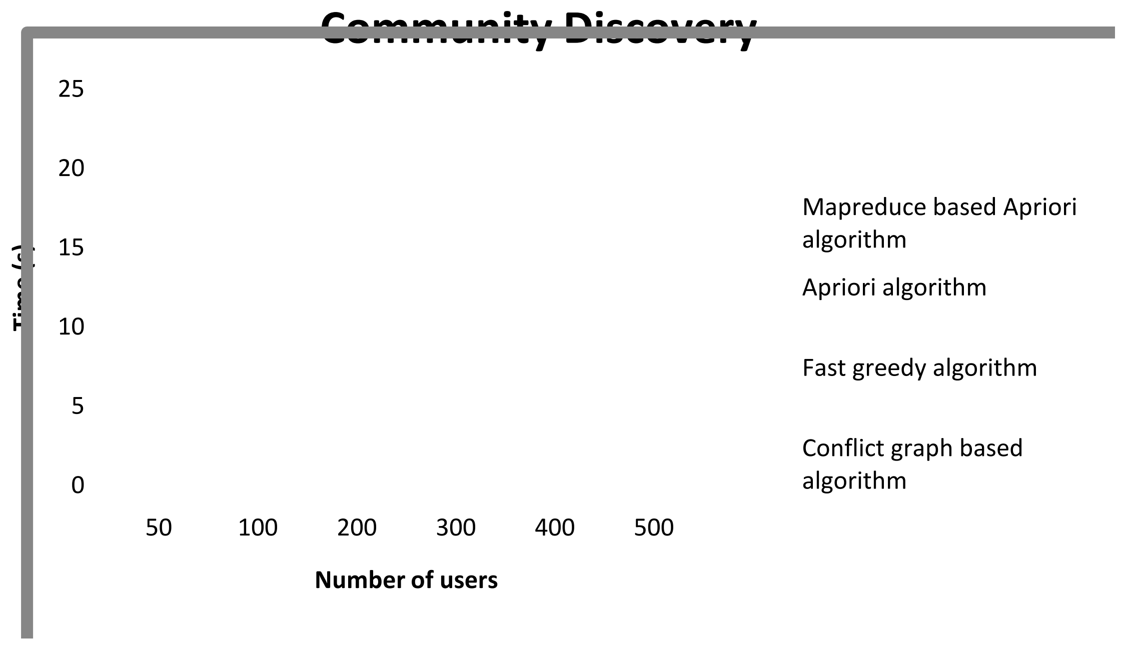

Figure 3: Comparison of execution time for community discovery algorithm with varying number of users

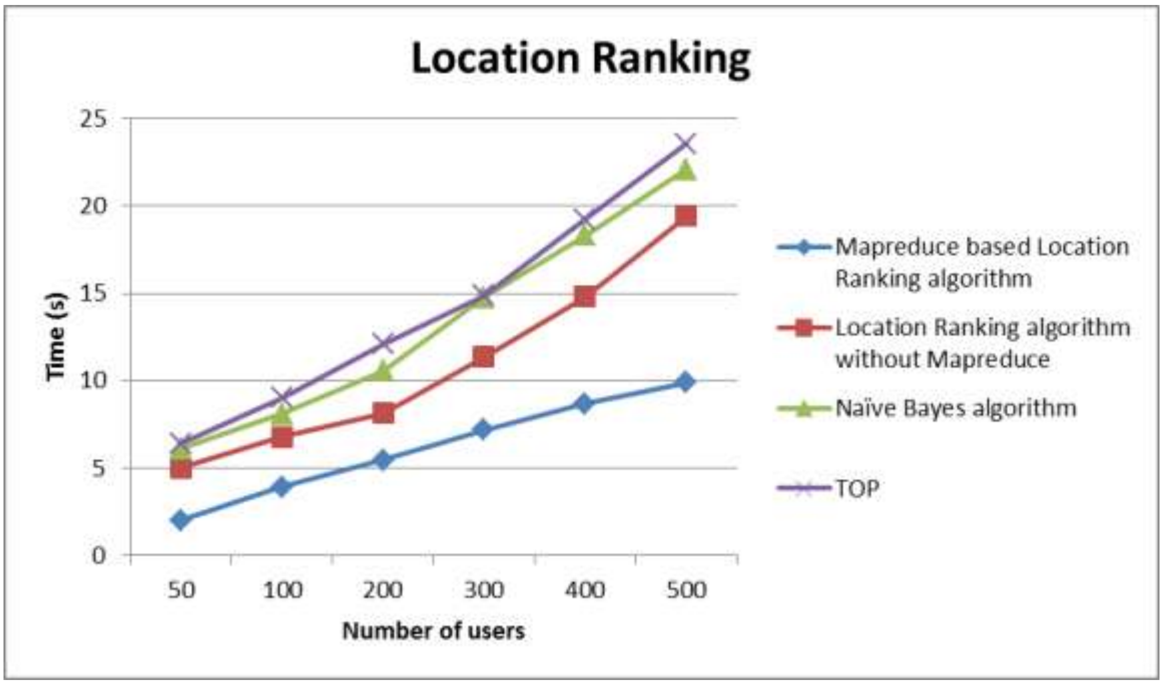

Figure 4: Comparison of execution time for location ranking algorithm with varying number of users

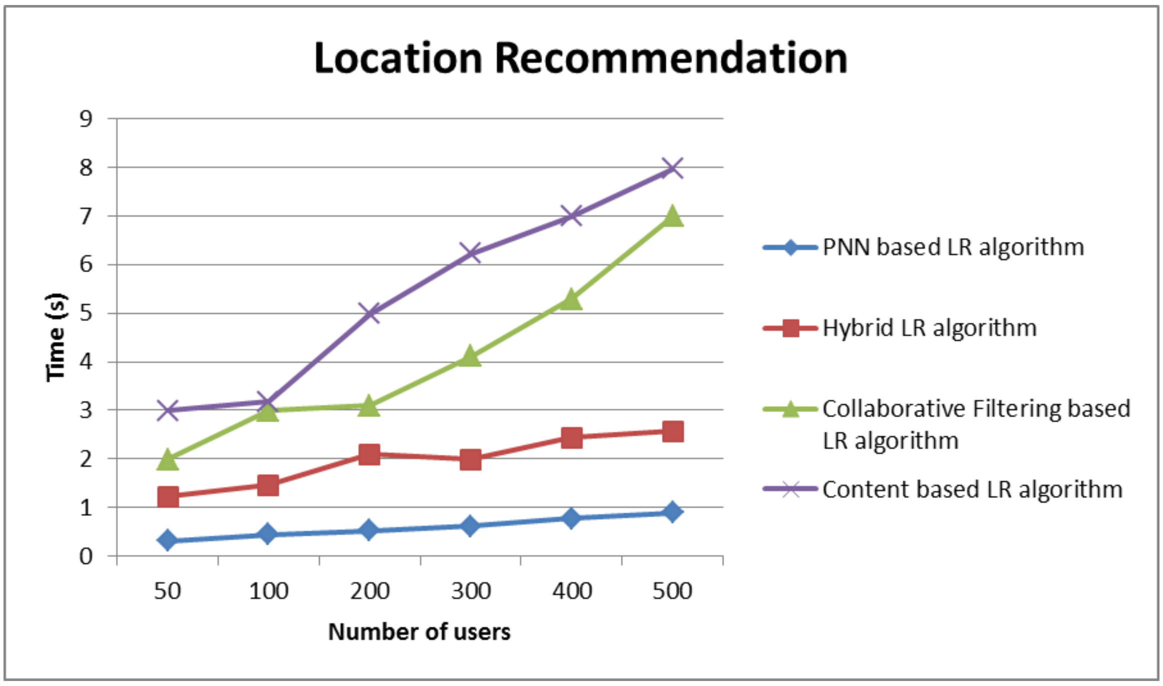

Figure 5: Comparison of execution time for location recommendation algorithm with varying number of users

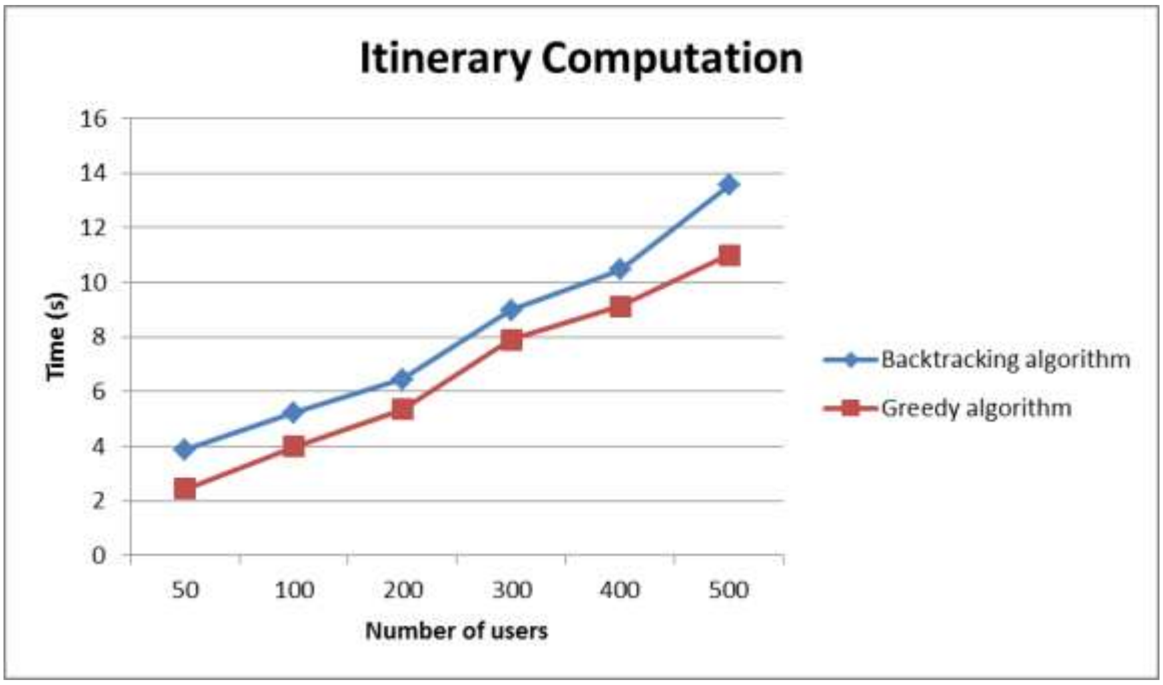

Figure 6: Comparison of execution time for itinerary computation algorithm with varying number of users

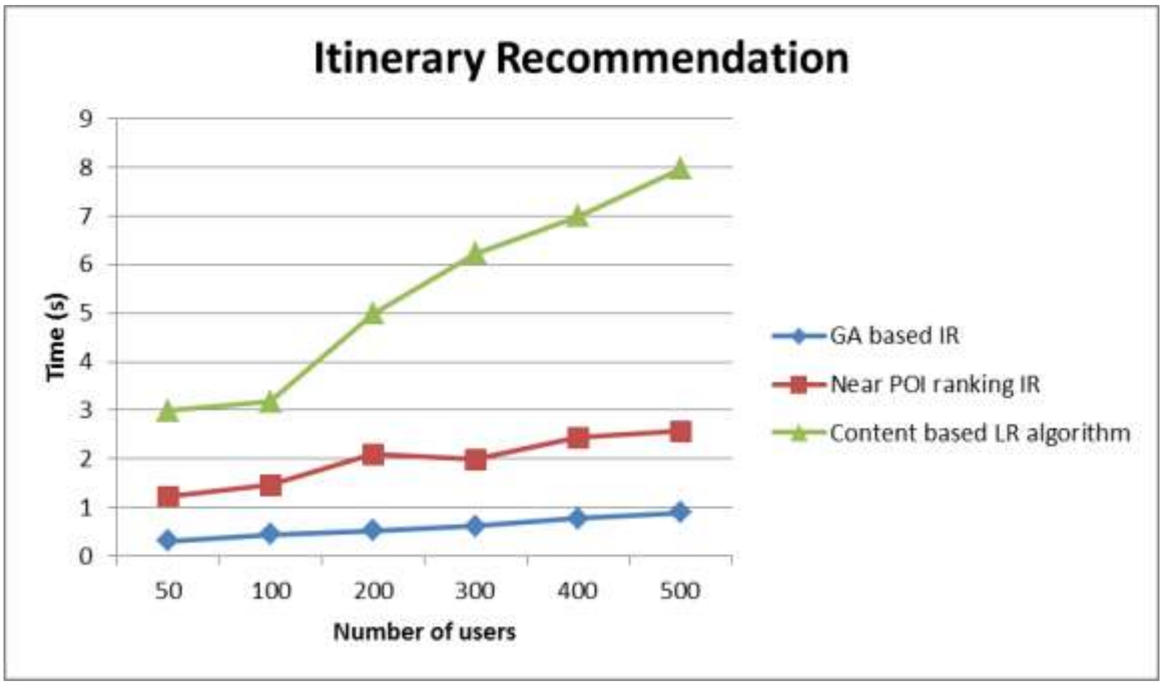

Figure 7: Comparison of execution time for itinerary recommendation algorithm with varying number of users

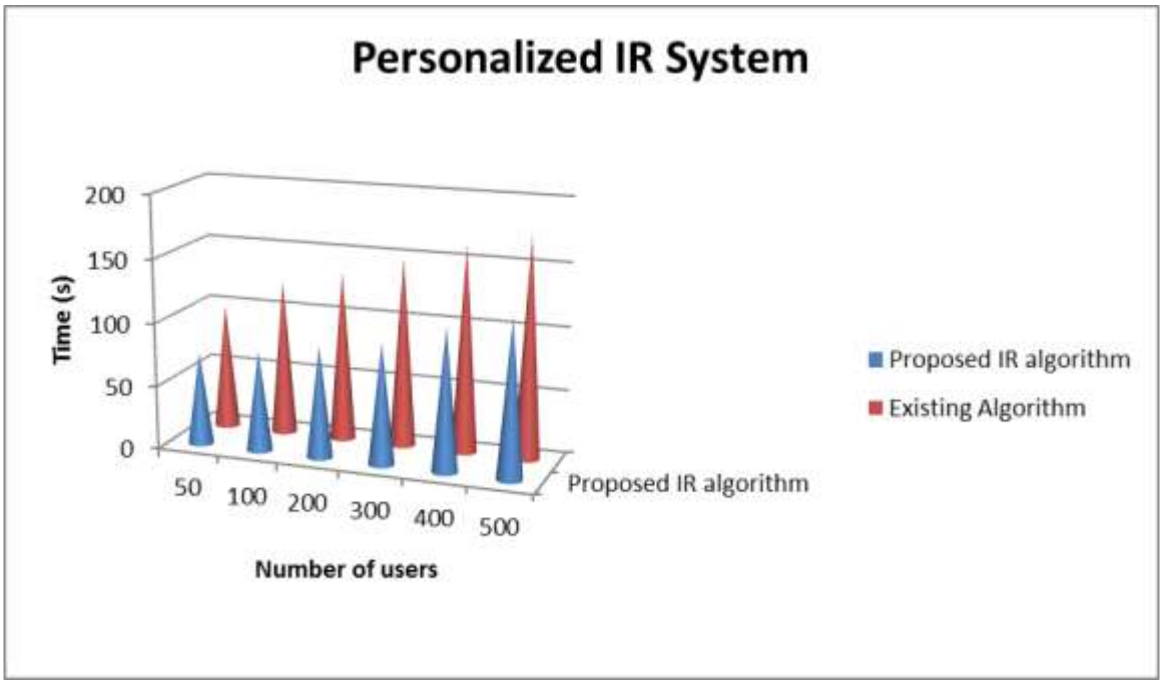

Figure 8: Comparison of execution time for itinerary recommendation algorithm with varying number of users

Figure 8 gives the overall system performance of the personalized itinerary recommendation system. The system gives better performance under varying number of users.

V CONCLUSION

This article presents a novel system for Location and Itinerary Recommendation on Location based Social Networks. The recommendation is done by considering the users' preferences and location history. The system also takes into account the tips left by the users who have already visited the locations. The system could give good recommendations in comparatively less response time.

ACKNOWLEDGEMENT

The authors would like to thank all the colleagues, scholars and supporters who have helped to make this work successful.

REFERENCES

[1] K. P. Madhu and D. Manjula, "A Community-Based Hybrid Location Recommendation System in Location-Based Social Networks", Asian Journal of Information Technology (AJIT), Vol.15, No. 07, 2016. (Accepted for Publication)

[2] Q. Li, Y. Zheng, X. Xie, Y. Chen, W. Liu and W.-Y. Ma, "Mining user similarity based on location history", Proc. Sixteenth ACM SIGSPATIAL international conference on Advances in geographic information systems, Irvine, CA, USA, November 05 - 07, 2008, Article No. 34.

[3] X. Xiao, Y. Zheng, Q. Luo and X. Xie, "Finding similar users using category-based location history", Proc. Eighteenth ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, ACM, New York, NY, USA, 2010, pp. 442-445.

[4] N. Eagle and A. Pentland, "Reality mining: sensing complex social systems", Journal of Personal and Ubiquitous Computing, Vol. 10, No. 4, 2006, pp. 255-268.

[5] N. Eagle and A. Pentland, "Eigenbehaviors: Identifying structure in routine", J. Behavioral Ecology and Sociobiology, Vol. 63, No. 7, April 2009, pp. 1057-1066.

[6] Y. Zheng and X. Zhou, Computing with Spatial Trajectories, Springer, 1 November 2011, Chapter 8: Location-based social networks: Users.

[7] M. Ye, P. Yin and W.-C. Lee, "Location recommendation for location-based social networks", Proc. Eighteenth SIGSPATIAL International Conference on Advances in Geographic Information Systems, ACM, New York, NY, USA, 2010, pp. 458-461.

[8] Ye, M., Yin, P., Lee, W., and Lee, D., "Exploiting geographical influence for collaborative point-of-interest recommendation", Proc. Thirty Fourth ACM International conference on Research and development in Information Retrieval (SIGIR), New York, NY, USA, 2011c, pp. 325-334.

[9] Z. Lixiao, X. Tingrong, L. Jie and L. Haiyan,"An Overlapping Community Detection Algorithm Based on Multistep Greedy Strategy", Proc. 4th International Conference on Intelligent Systems Design and Engineering Applications, IEEE, 2013.

[10] P. Singh, A. Chakraborty, and B. S. Manoj," Conflict Graph based Community Detection", Proc. 8th International Conference on Communication Systems and Networks (COMSNETS), IEEE, 2016.

[11] S. Gouthami, G. J. Mary and P. S. Rao," Ranking Popular Items By Naive Bayes Algorithm", International Journal of Computer Science & Information Technology (IJCSIT) Vol 4, No 1, Feb 2012.

[12] M. D.Choudhury, M.Feldman, S. A.Yahia, N. Golbandi, R.Lempel and C.Yu, "Automatic Construction of Travel Itineraries using Social Breadcrumbs", HT'10, Toronto, Ontario, Canada, June 13–16, 2010.

[13] S.Dunstall, M. E. T.Horn, P. Kilby, M. Krishnamoorthy, B. Owens, D. Sier, and S. Thiebaux," An Automated Itinerary Planning System For Holiday Travel", Journal of Information Technology & Tourism, Vol. 6 pp. 195–210, Cognizant Comm. Corp., 2004.

[14] Nirmala and A. AmbethRaja "An Improved Travel Package Recommendation System Using (Cope)", Weekly Science Research Journal, Volume-2 | Issue-46 | 28 May-2015, ISSN 2321-7871 Impact Factor: 1.4210 (UIF) [Yr. 2013].

[15] S. B. Roy, G. Das, S. A. Yahia and C. Yu, "Interactive Itinerary Planning", Proc. IEEE 27th Int'l Conf. Data Eng. (ICDE), pp. 15-26, 2011.

[16] P. S. Ghogare and H. K. Khanuja, "Customized Travel Planner using MapReduce and Approximation Algorithm", International Journal of Computer Applications (0975 – 8887), Volume 119 – No.23, June 2015.

[17] H. T. Chang, Y. M. Chang and M. T. Tsai, "ATIPS: Automatic Travel Itinerary Planning System for Domestic Areas", Journal of Computational Intelligence and Neuroscience, Volume 2016, Article ID 1281379, Hindawi Publishing Corporation.

[18] C. A. Reddy and V. Subramaniyaswamy," An Enhanced Travel Package Recommendation System based on Location Dependent Social Data", Indian Journal of Science and Technology, Vol 8(16), July 2015.

BIOGRAPHY

MADHU K P is the corresponding author of this paper. He is currently working as senior Assistant Professor in Computer Science and Engineering at Government Engineering College, Kottayam, Kerala, India. He is pursuing his Ph.D. in the Faculty of Information and Communication Engineering, Anna University, Chennai, India. He received his M.E. degree in Computer Science and Engineering from PSG College of Technology, Coimbatore, India in the year 2010 and B. Tech. degree in Computer Science and Engineering from Mahatma Gandhi University, Kottayam, India in the year 2001. His fields of interests are Location Based Social Networks, Social Network Analysis, Cloud Computing, Virtualization Techniques, Data Mining etc. He has published 5 papers in national/International conferences and journals.

Dr. D. MANJULA is currently working as a Professor in the Department of Computer Science and Engineering in Anna University, Chennai, India. She received her Ph.D. degree in the Faculty of Information and Communication Engineering from Anna University, Chennai, India in the year 2004, M.E. degree in Computer Science and Engineering from Anna University, Chennai, India in the year 1987 and B.E. degree in Electronics and Communication Engineering from Thiagarajar College of Engineering, Madurai, India in the year 1983. She has published three books. Her present research interests include Cloud Computing, Virtualization Techniques, Information Retrieval, NLP, Text Mining, Parallel Processing, Grid Computing and Databases. She has published more than 200 papers in National/International Conferences and Journals.