

Social Network Analysis and Mining

Mining Optimal Meeting Points for Moving Users in Spatio-Temporal space

--Manuscript Draft--

Manuscript Number:	SNAM-D-18-00050R1
Full Title:	Mining Optimal Meeting Points for Moving Users in Spatio-Temporal space
Article Type:	Original Article
Corresponding Author:	Sonia Khetarpaul Shiv Nadar University Delhi, INDIA
Corresponding Author Secondary Information:	
Corresponding Author's Institution:	Shiv Nadar University
Corresponding Author's Secondary Institution:	
First Author:	Sonia Khetarpaul, Ph.D.
First Author Secondary Information:	
Order of Authors:	Sonia Khetarpaul, Ph.D. S K Gupta L Venkat Subramaniam
Order of Authors Secondary Information:	
Funding Information:	
Abstract:	Scheduling a meeting is a difficult task for people who have overbooked calendars and often have many constraints. This activity becomes further complex when the meeting is to be scheduled between parties who are situated in geographically distant locations of a city and have varying traveling patterns. To achieve this, we first propose a solution to determine optimal meeting location for two moving users in the Euclidean space. Then, We generalize the problem by considering variable number of moving users and evaluate optimal meeting point (OMP) on the road network. We extend the work of Yan et al. [1] in this domain by incorporating some real life constraints like variable number of users, varying travel patterns, flexible meeting point and considering road network distance. Experiments are performed on a real-world dataset and show that our method is effective in stated conditions.
Response to Reviewers:	Please see the attachment.

[Click here to view linked References](#)

Noname manuscript No. (will be inserted by the editor)
--

Mining Optimal Meeting Points for Moving Users in Spatio-Temporal space

Sonia Khetarpaul
S K Gupta · L Venkata Subramaniam

the date of receipt and acceptance should be inserted later

Abstract Scheduling a meeting is a difficult task for people who have overbooked calendars and often have many constraints. This activity becomes further complex when the meeting is to be scheduled between parties who are situated in geographically distant locations of a city and have varying traveling patterns. To achieve this, we first propose a solution to determine optimal meeting location for two moving users in the Euclidean space. Then, we generalize the problem by considering variable number of moving users and evaluate optimal meeting point (OMP) on the road network. We extend the work of Yan et al. [1] in this domain by incorporating some real life constraints like variable number of users, varying travel patterns, flexible meeting point and considering road network distance. Experiments are performed on a real-world dataset and show that our method is effective in stated conditions.

Keywords spatio-temporal data · meeting point · search space pruning

1 Introduction

Recent advances in wireless communication and positioning devices like Global Positioning Systems (GPS) have generated significant interest in the field of analyzing and mining patterns present in spatio-temporal data. The pervasiveness of location-acquisition tech-

nologies (GPS, GSM networks, etc.) has enabled convenient logging of location and movement histories of individuals. The increasing availability of large amounts of spatio-temporal data pertaining to the movement of users has given rise to a variety of applications and also the opportunity to discover travel patterns. Managing and understanding the collected location data are two important issues for these applications.

The amount of data generated by such GPS devices is large. For example, most GPS devices collect location information for a user every 2 to 5 seconds [2]. This means that for a single user between 17000 to 44000 data points are generated in a single day. Aggregated over tens of users over several days the data size grows exponentially [2,3]. This is extremely rich data and a lot of useful analysis can be performed on this data, potentially giving rise to a variety of application.

Optimal Meeting Point queries (or OMP queries) are useful in many real world applications, ranging from location-based services to computer games. For example, a travel agency may run an OMP query to decide the location for a tourist bus to pick up the tourists, so that the tourists can make the least effort to get to the meeting point. OMP queries are also important for merging military forces in a war field or finding a place that is convenient for military officers to hold a meeting [4]. In strategy games like Warcraft1, a computer player may need to find OMPs. In location based social networks, different users connected in the network can plan a get together and may run an OMP query to determine a convenient meeting location for all.

We solve the optimal meeting point problem in two different ways. In the first part we extend our previous work of interesting location determination [5] and determine the optimal meeting point for two users in Euclidean space. We create the travel graph of a user

Department of Computer Science and Engineering
SNU, Gautam Buddha Nagar, UP, India
E-mail: sonia.khetarpaul@snu.edu.in
Department of Computer Science and Engineering
IIT Delhi, India
E-mail: skg@cse.iitd.ac.in
IBM Research-India
E-mail: lvsubram@in.ibm.com

by connecting their stay points and mine these travel graph to determine optimal location for meeting. In the second part we solve the optimal meeting point problem for variable number of moving users and consider road network distance instead of euclidean distance. We extend the work of Yan et al. [1] in this domain by incorporating some real life constraints like variable number of users, varying travel patterns, flexible meeting point and considering road network distance.

1.1 Literature Review

In this paper we addresses the problem of determining optimal meeting point from the historical GPS traces of moving users on road network. Our approach consists of two parts, first to mine historical traces of users and predict their locations points on road network and second is to determine optimal meeting point of moving users on road network. Based on this related work consists of two parts: spatio-temporal data analysis and optimal meeting point determination.

spatio-temporal data analysis: There has been a lot of prior work on using spatio-temporal data to track movement history. There has also been work around integrating multiple users' information to learn patterns and understand a geographical region. GeoLife is a location-based social-networking service on Microsoft Virtual Earth. GeoLife enables users to share travel experiences using GPS trajectories [6,2,7,8]. It finds the top most interesting locations, classical travel sequences in a given geospatial region. In geolife to find out interesting locations in a given geospatial region HITS model (Hypertext induced topic model search) is introduced. The paper [9] is based on Hybrid Prediction Model, which estimates an object's future locations based on its pattern information as well as existing motion functions using the object's recent movements. Other systems like and CityVoyager [10] are designed to recommend shops and restaurants by analyzing multiple users' real-world location history. Mobile tourist guide systems [11, 12,10,6] typically recommend locations and sometimes provide navigation information based on a user's real-time location. In contrast, our approach is based on assumption that users are moving not stationary, and follows a regular routine during weekdays. Our approach to determine users location points is based on simple statistics that applied on users historical GPS traces.

Optimal Meeting Point: We first give an overview of the related work for OMP queries in Euclidean space and in road networks.

Optimal Meeting Point problem is well studied in different forms in Euclidean space [13–16] during 60s70s.

When the Euclidean distance is used to measure the distance, the minimum-sum OMP query is called the Weber problem [13], and the minimum sum OMP is called the geometric median of the location point set. Cooper in 1968 [13] extended the Weber problem by formalizing the problem of minimizing the weighted sum of powers of the Euclidean distances. However, it is shown that no closed form formula exists for the solving minimum-sum OMP query and its generalizations, and these problems are usually solved by gradient descent methods [17,18].

Aggregate Nearest Neighbor(ANN) queries [19–21] are closely related to our OMP queries. However, the fundamental difference is that, for Aggregate Nearest Neighbor queries, the result location is chosen among a finite data location point set $L = \{l_1, l_2, \dots, l_n\}$, while for OMP queries, the result location is chosen from a spatial geographical area that contains infinite number of points. Like various nearest neighbor queries [19,20, 22,23], the OMP query is also important for spatio-temporal databases.

On the other hand, the OMP query is not well studied in terms of road networks, where the road network distance is taken up as the distance metric. Recently, Xu et al. [24] proposed a solution to this problem by checking all the split points on the road network. It is proved in [24] that an OMP must exist among the split points. So, they propose an algorithm that checks the split point of each location point in Q on each edge in the road network $G = (V, E)$, and picks the split point with the smallest sum of network distances as the OMP. As a result, the search space is $|Q| \cdot |E|$, which is huge. Although [24] includes a pruning technique to skip some split points that are guaranteed not to be an OMP, the search space after pruning is still very large. Therefore, a novel road network partitioning scheme is proposed in [24] to further prune the search space, based on the property that the OMP is strictly confined within the partition where all the objects in the query set Q are located. After that, [1] proves that an OMP must exist either on vertex or on query points, there is no need to check all the split points. So search space is reduced to $|Q| + |V|$. To further reduce the search space, two phase convex-hull-based search space pruning techniques are proposed in [1]. In contrast, our approach considers that user are moving not stationary, so we are predicting user locations, the directions in which they are moving and pruning the search space based on their locations before and after the meeting. Our approach considers both spatial and temporal aspect of data. This paper is extension of our previous research [25] and [26].

Our approach to determining OMP for two users in Euclidean space is defined in section 2 and its data analysis and results are discussed in section 3. And, the

OMP problem for a variable number of users on the road network is discussed in section 4 and its analysis and results are discussed in section 5.

2 Optimal Meeting Point for Two Users in Euclidean Space

In this section, we apply statistical and algebraic operations and determine the optimal meeting point for two users in Euclidean space. We start by extending our previous work [5] and determine stay points and interesting locations.

A stay point [5] represents a geographical location where an individual spends a significant amount of time within a pre defined time interval. It can be a work place, home, restaurant, historical place, congestion area, shopping mall, a stadium, worship places, etc. Stay points are determined for individuals based on their GPS traces. Given the time stamped GPS log of an individual, we use this information to determine this person's stay points at different time intervals. Since the log contains both location and time information it is possible to determine how much time a person spent at a particular location in a given time period.

An interesting location [5] $IntLoc(T_{ij})$ is one that is visited by many individuals during time interval t_i to t_j . $IntLoc(T_{ij})$ is a geographical region where the number of stay points of distinct users within a time interval T_{ij} exceeds a given $ThresCount$. $ThresCount$ is a user defined parameter. An interesting location can be a historical place, a good restaurant, a shopping complex, a stadium, a garden or any place that is visited by many users during the same time interval.

We start by creating trajectories for each user from their given GPS logs. First we determine the temporal interesting locations by mining the bag of stay points of users during different time intervals. Then, we create the travel graph for each user and mine two travel graphs to determine optimal meeting location or point.

In the following subsections, we define the travel graph and method to construct it.

2.1 Connecting Stay Points to Obtain Users' Travel Graphs

For a given user the GPS trajectory over several days is a series of criss-crossing lines from which it is very difficult to extract useful information.

We define the **Travel Graph** G of a user as the sequence of stay points connected by directed edges representing the user trajectory. We draw the travel graphs for specific time windows. People usually follow similar

cyclic patterns in their travel schedules. Therefore for a given user her week day schedules may be similar but different from her weekend schedules. The Travel Graph is then the most common path usually followed by the user for the given day and time. This travel graph is obtained for a given user by combining data from several days or even several months. Algorithm 1 gives method for determining the Travel Graph for a user. It takes the GPS Log of a user and first determines the stay points of the user. Then the mode operation is applied to determine the most frequent trajectories. Unnecessary nodes and edges are pruned based on the given meeting time.

Algorithm 1: Travel Graph Algorithm

Data: GPS Log, date and Time.

Result: A Graph $G = (V, E)$ where V is the set of Vertices represents the stay Points and E is set of directional edges representing the user trajectory.

begin

```

    MeetingMonth  $\leftarrow$  date.getMonth()
    Offset  $\leftarrow$  1
    ProcessMonths  $\leftarrow$  Month - Offset
    StartTime  $\leftarrow$  time.hours - Offset
    EndTime  $\leftarrow$  time.hours + Offset
    GPS[n]  $\leftarrow$ 
    extractGPSLog(U, date, ProcessMonth,
    StartTime, EndTime)
    Su[m]  $\leftarrow$ 
    SPCalculate(GPSU, ThresDistance, ThresTime)
    modeGPSU[x][2]  $\leftarrow$  mode(GPSU)
    i  $\leftarrow$  0
    while i < x do
        if modeGPSU[x][1] > ThresValue then
            frequentGPSU[i]  $\leftarrow$  modeGPSU[i][0]
            i  $\leftarrow$  i + 1
    i  $\leftarrow$  0, j  $\leftarrow$  0
    while i < frequentGPSU.length do
        while j < m do
            if Round(frequentGPSU[i], 3) = Su[j]
                then
                    ReplaceAllOccurrence(frequentGPSU[i],
                    Su[j])
                j  $\leftarrow$  j + 1
            i  $\leftarrow$  i + 1
    Return(frequentGPSU)

```

2.2 Identifying the Common Meeting Point

Travel graphs of users can be created offline. First, from the given two users' GPS logs and meeting date we can evaluate their most frequent and recent locations

trajectories. These locations trajectories represent the edges of graph. Now stay points evaluated earlier are inserted to interconnect these edges. These stay points form the nodes of graph. Next step is to minimize the graph. For this, a time interval is evaluated from the input meeting time and unnecessary nodes and edges are pruned from the graph using this time interval. The resulting output is a pair of minimized directed travel graphs.

Algorithm 2 takes these two minimal directed graphs as input and determines the rankwise list of possible meeting places for two users. In this algorithm, directions, timings and stay points of users are important parameters. Different cases arise based on these parameters. To determine meeting locations for two users we calculate the minimum distance GPS points from the two directed travel graphs of the users. A list of GPS points sorted on minimum distance is obtained. A threshold time difference *ThreshTime* is set to 30 minutes. From the sorted list, for every pair of GPS points their time difference *TimeDiff* of visiting that points are calculated. If the distance is minimum, *TimeDiff* is below *ThreshTime* and any one of GPS point is stay points of the user then that stay points is ranked higher as a meeting place. But if nearest GPS points are not the stay points of any user than we search a nearby stay point from the list of interesting location during that time interval and this place can be a meeting place and ranked lower than the previous case. A minimal distance GPS traces with *TimeDiff* more than *ThreshTime* are removed.

Given two users u_1 and u_2 , with GPS points n_1 and n_2 respectively. The complexity of our algorithm is $p \times \ln(p)$ where $p = (n'_1 \times n'_2)$ and $n'_i = n_i/f$ where f is the compression factor due to the truncation. The longitude and latitude values of each GPS point are truncated so that it points to the defined geographical region. For example, truncating to three decimal places corresponds to a physical distance of about 100 meters.

The ranking is done as per the following use cases:

Case 1: Two users have intersecting stay points (nodes) with the time difference less than *ThreshTime* as shown in Figure 1(a), in this case the meeting place can be their common stay point.

Case 2: When two users have intersecting stay points (node) and trajectory (edge) with the time difference less than *ThreshTime* as shown in Figure 1(b), in this case meeting place can be the intersecting stay point.

Case 3: When two users have intersecting trajectories (edges) with the time difference less than *ThreshTime* as shown in Figure 1(d). In this case a list of interesting places are considered as input and a nearest

Algorithm 2: Finding the Common Meeting Point

Data: (G_{U_i}, G_{U_j}) - GPS Log of U_i and U_j , Date and Time Of Meeting, List of Interesting Locations(IntLoc).

Result: Ranked List of Meeting Places.

```

begin
  GraphUi[EV1] ← BuildGraph(GUi, date, time)
  GraphUj[EV2] ← BuildGraph(GUj, date, time)
  i ← 0
  j ← 0
  while i < EV1 do
    while j < EV2 do
      DistanceNodes[i][2] ←
        Distance(GraphUi[i], GraphUj[j])
      DistanceNodes[i][0] ← GraphUi[i]
      DistanceNodes[i][1] ← GraphUj[j]
      j ← j + 1
    i ← i + 1
  Sort(DistanceNodes[i][2])
  k ← 0
  rank ← 0
  while k < DistanceNodes.length() do
    if DistanceNodes[k][2] = 0 then
      if TimeDiff(DistanceNodes[i][0],
        DistanceNodes[i][1]) ≤ ThreshTime
      then
        DistanceNodes[k][3] ← Rank + 1
      else if DistanceNodes[k][2] ≠ 0 then
        DistanceNodes[k][2] ←
          DistanceNodes[k][2]/2
        if TimeDiff(DistanceNodes[i][0],
          DistanceNodes[i][1]) ≤ ThreshTime
        then
          DistanceNodes[k][3] ← Rank + 1
      k ← k + 1
  Loc ←
  NearInterestingLoc(IntLoc, DistanceNodes)
  Return(DistanceNodes, Loc)

```

interesting location **IntLoc** is chosen as the meeting place.

Case 4: When two users do not have any intersection points (non-intersecting graphs) as shown in Figure 1(c). In this case the two nearest GPS points in their trajectories are determined, with the time difference less than *ThreshTime*. A nearest interesting location **IntLoc** closest to the center of the line connecting these GPS points is chosen as the meeting place.

3 Data Analysis and Results for Two Users

In this section, we first briefly describe the GPS dataset [2] used for analysis. Next we present the results of applying our approach for determining a rank-wise list of meeting places.

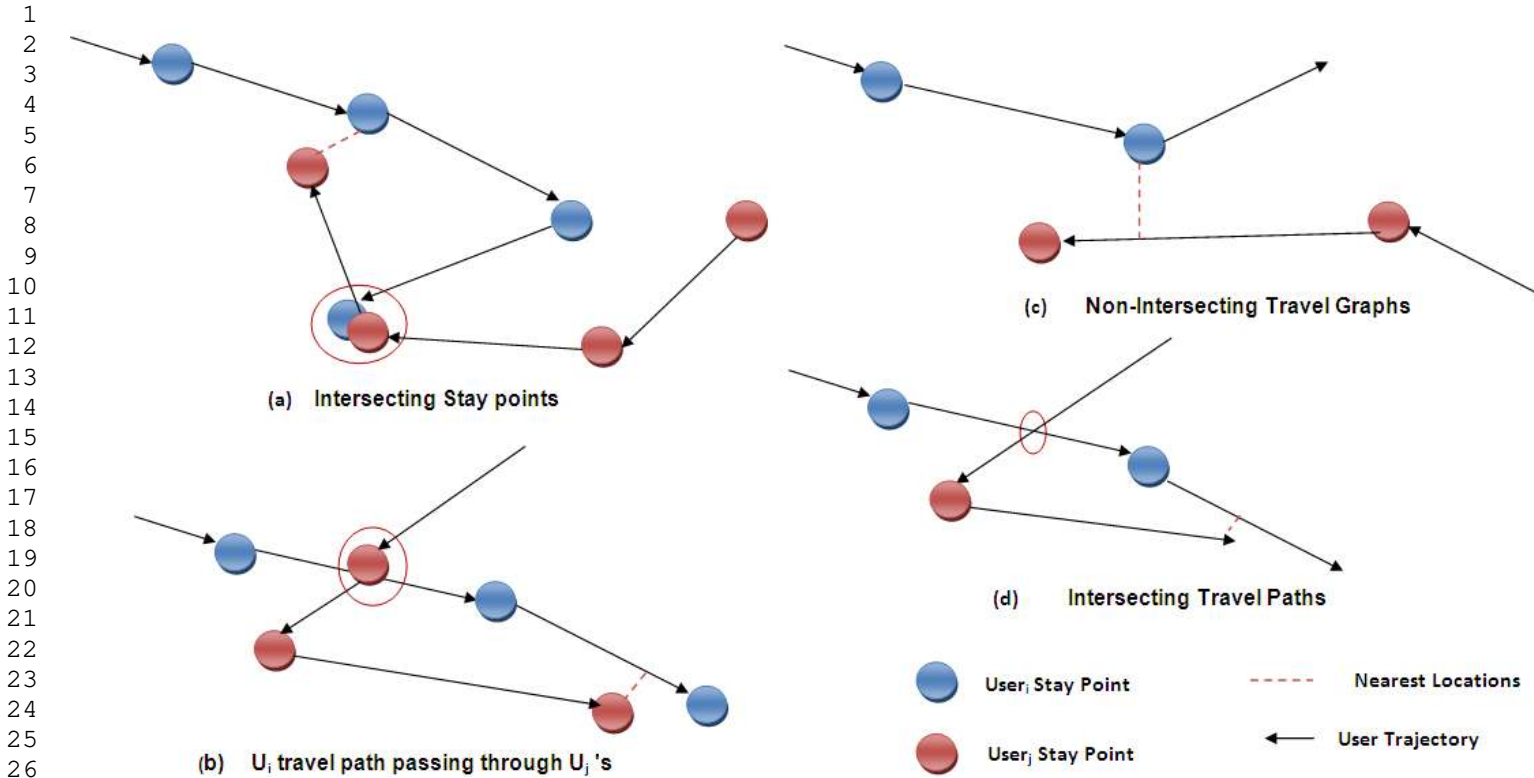


Fig. 1: Travel Graph use cases for two users

3.1 GPS Trajectory Dataset

We worked on 126 users GPS traces from the Geolife dataset. This subset consists of a total of 68612 days data with 5,832,020 GPS points. The total area covered by the GPS logs exceeded 3,880,951 Sq. kilometers. The majority of the data was created in Beijing, China.

3.2 Building Travel Graph

To build a travel graph from input GPS logs the most frequent GPS trajectories are extracted and for this a threshold value 10 is set. If a user visits a GPS point more than 10 times during weekdays within a month then only those GPS points are extracted and others are removed. For each such GPS point the typical time during the day when the user is there is also determined.

We obtain a directed graph where the nodes are stay points. The graphs of a user for different times of the day are different. For example at 8 am the user may typically be at home and start traveling to her office that she reaches at 9 am. GPS traces and stay points between $Time - Offset$ and $Time + Offset$ are evaluated for User U_i to construct the Travel Graph.

3.3 Meeting Locations Determination

To determine meeting locations for two users we calculate the minimum distance GPS points from the two directed travel graphs of the users. The travel graphs are constructed for the meeting time to determine the exact travel path of the user and fix the meeting based on this. In Table 1, we have shown the meeting point between two random users in our database, with different given meeting times. For example, in first case of Table 1, we would like to arrange a meeting between these two users at 13:00 hrs. We construct the travel graph for these users by considering their stay points between 12:00 hrs to 14:00 hrs. Then we determine the best locations for their meeting. Three meeting locations have been suggested here. The first four columns give the (latitude, longitude) of each user that is closest to the meeting point at this time. $MinDist$ is the distance between the users from these points. T_1 is the time when the user is at this stay points. In the column SP_1 value 1 means that this (latitude, longitude) is a stay points for this user. The last three columns give the meeting point based on our meeting point algorithm and given meeting time for each case.

The Naive approach to arranging meetings is to find the center point of two people's locations before the

meeting time. So if user 1 is likely to be in the office at 13:00 and user 2 is likely to be in another office at the same time. Then their meeting point would be some point in between their offices. We have used historical data to determine where the user is likely to be around the meeting time, and also determined his travel path to take into account the direction of his travel. We have compared our approach to a baseline naive approach.

Similarly, in the other cases listed in Table 1, we determined the travel graphs of the users and mined these travel graphs for a given time interval based on meeting time and mined two graphs to determine rank wise meeting locations.

3.4 Comparison with Baseline Approach

We have evaluated meeting places for randomly selected 15 user pairs for different timing within a day. In baseline approach all week days GPS traces of given two users for 15 days are extracted from GPS logs. Along the given meeting time a time window of 6 hours is considered and minimum distance GPS points are determined. This is close to the naive approach described above. It is observed that baseline approach gives minimum distance GPS points but user arrival time difference at that meeting place is very large. Also the arrival time to these GPS points are very different from meeting time. For 15 pairs of users, we found that our approach gave an average time difference of reaching the meeting point to the actual time of meeting to be 19 minutes, whereas the baseline approach gave this time to be 93 minutes. Also in 11 of the cases using our approach the meeting place found was the stay points of one of the users. However, for the baseline approach this number was 5.

4 Optimal Meeting Point for Variable Number of Users on the Road Network

In this section, we solve the problem to determine optimal meeting point for a variable number of moving users on the road network. Again, we assume that users have varying traveling patterns which can be discovered from logs of their travel history. These logs are generated by standard GPS devices. We assume that these devices generates traces at the same rate and in same format(same granularity) or GPS traces can be transformed to a fixed format. From these GPS traces, we create *Trajectories*.

Let $\{u_1, u_2, \dots, u_n\}$ be a set of n users. It is noted that if all of these n users wish to meet at some fixed time, and the meeting point is desired as *OMP*, then

the sum of distances would be the total cost of meeting(for all the users). It is proposed to minimize this cost. Formally, optimal meeting point for group of people is defined as:

Optimal meeting point is defined to be

$$\arg \min_{OMP \in N} [\sum d_N(u_i(t), OMP)]$$

where $d_N(x, y)$ is the shortest distance between two point x and y on the road network N .

Informally, we define the **optimal meeting point** as a geographical point on the road network where the sum of distances traveled by all the users is minimum. In Figure 2 total distances traveled by all users is minimum at point *OMP*, and is 23 K.M.

We introduce two measures to evaluate the processing cost i.e. the minimum-sum-center and the direction of movement. These measures operate over the spatio-temporal domain of each moving user by applying a network distance to all users tracked. Each measure induces a spatio-temporal relation that minimizes or maximizes a property over the underlying network graph for the given measure and the given set of moving users. We develop query processing algorithms for computing the value of these measures and to determine spatio-temporal relations and the point on the road network that yields the optimal value of relation's value from the predictive graph of moving users. Finally, we demonstrate how users movement histories and projected movement trajectories can be used to determine the optimal meeting point. We also consider predicted directions of motion of users at time t and at time $t + \delta t$ to optimize the total distance covered by each user. It is considered by examining the consecutive hulls of location points.

GPS and other positioning devices generate location information every few seconds (often at the interval of two to five seconds). An individual carrying such a device potentially generates thousands of GPS points everyday. We aggregate all the data from multiple users and predict the location of each meeting user at given point of time. After predicting the location points for the user optimal meeting point is determined.

Now, we define the terms and notations used for predicting location points:

Grid: Grid divides the whole geographical area into $k = m * p$ cells and represented by $C = \{c_1, c_2, \dots, c_k\}$, where m is number of unique δLat and p is number of unique δLng .

Cell: A cell c_i is a rectangular element of grid C dividing the region of interest. They are sequenced major rowwise and minor columnwise.

Table 1: Ranked list of meeting places for two users

Sno	Lat_1	Lng_1	Lat_2	Lng_2	T_1	T_2	SP_1	SP_2	MinDist	Rank	Meeting Lat	Meeting Lng	Meeting Time(hrs)
1	39.977	116.331	40.002	116.323	12:44:58	12:33:02	1	0	2.862	1	39.977	116.331	13:00:00
	39.977	116.328	40.002	116.323	12:44:04	12:33:02	0	0	2.812	2	39.982	116.326	13:00:00
	39.977	116.331	40.002	116.322	12:44:58	13:14:53	1	0	2.883	3	39.977	116.331	13:00:00
2	39.975	116.331	39.975	116.331	10:07:11	10:34:57	0	1	0	1	39.975	116.331	10:00:00
3	39.975	116.331	39.976	116.331	11:14:29	11:30:46	1	1	0.111	1	39.975/ 39.976	116.331	12:00:00
	39.975	116.343	39.976	116.331	11:10:26	11:30:46	0	1	1.0285	2	39.976	116.331	12:00:00
4	39.938	116.337	39.975	116.33	13:59:26	13:55:10	0	1	4.157	1	39.975	116.33	14:00:00
	39.94	116.348	39.975	116.33	13:42:09	13:55:10	0	1	4.183	2	39.975	116.33	14:00:00
5	22.295	114.175	22.282	114.184	17:02:46	17:32:38	0	0	1.716	1	22.2885	114.1795	17:00:00
	22.295	114.174	22.282	114.184	17:04:38	17:32:38	0	0	1.774	2	22.2885	114.179	17:00:00
	22.296	14.176	22.282	114.185	17:00:54	17:30:20	0	0	1.811	3	22.289	114.1805	17:00:00
6	39.94	116.348	39.973	116.332	13:42:09	14:06:53	0	0	3.914	1	39.9565	116.340	14:00:00
	39.938	116.343	39.973	116.332	13:50:32	14:06:53	0	0	4.003	2	39.9555	116.3375	14:00:00
7	39.975	116.331	39.975	116.327	11:14:29	11:24:12	1	0	0.340	1	39.975	116.331	11:00:00
	39.968	116.428	39.971	116.423	11:00:07	11:22:32	0	0	0.541	2	39.9695	116.4255	11:00:00
	39.968	116.428	39.971	116.419	11:00:07	11:00:19	0	0	0.8363	3	39.9695	116.4235	11:00:00

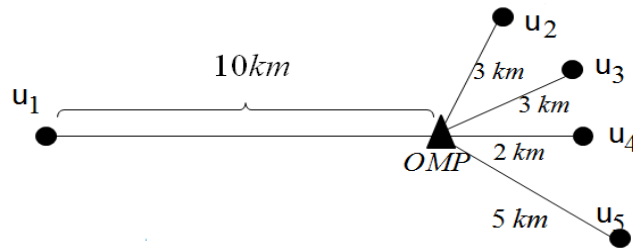


Fig. 2: Optimal Meeting Point

Road Network Distance: Road network distance is the shortest length of path between two cells on road network. This distance is obtained using function $d_N(c_i, c_j)$, it returns the length of path between two cells of grid C , c_i and c_j on road network. This distance function $d_N(c_i, c_j)$ can be realized through Google maps.

Location Point: A location point l_i represents the location of the i^{th} user on grid C at a point of time.

We determine the location points of all individual users at given point of time. Location point of a user is a predicted geographical location where the user is, at given meeting time. This prediction is done by statistical analysis of their past *Trajectories*.

4.1 Location Point Determination

Most of us generally follow a specific travel pattern during working days. To determine the location point for each individual user, at a given point of time, we analyze their past GPS logs. By applying statistical op-

erations on their past GPS trajectories, we are able to predict their locations at a given point of time. For location points analysis w.r.t. to time and space, the whole geographical space is divided into a grid, where each cell $c(l*w)$ represents a small geographical region and is assigned a number. Twenty four hours in a day are divided into small time periods of length δt . The log records are mapped on to this grid. For each user, his/her location cell number, after every δt interval of time is identified from his/her log records. User locations of many days at different time intervals are summarized to generate his/her spatio-temporal graph.

We determine the maximum and minimum value of latitude and longitude ($Ar = (maxlat - minlat) * (maxlng - minlng)$ gives total area of city), which define our domain of interest. $\delta lat * \delta lng$ forms the area of a single cell within the grid. They are sequenced major rowwise and minor columnwise, from 1 to K , where K is maximum number of cells. Location of each user is predicted in terms of a cell number. Mapping of user's GPS location into cell number and cell number into GPS location is done using following conversions.

1 **Mapping given location (Lat_i, Lng_i) to cell num-**
 2 **ber:**

$$3 \quad Cellno = ((Lng_i - minLng)/\delta Lng) * \text{No. of unique}$$

$$4 \quad \delta cLng + ((Lat_i - minLat)/\delta Lat)$$

6 **Mapping given cell number to location (Lat_i, Lng_i)**

$$7 \quad Cellno = lng_{ind} * \text{No. of unique } \delta lng + lat_{ind}$$

8 Where lng_{ind} is quotient and lat_{ind} is remainder
 9 when $cellno / (\text{No of unique } \delta lng)$

$$10 \quad Lat_i = minLat + lat_{ind} * \delta Lat$$

$$11 \quad Lng_i = minLng + lng_{ind} * \delta Lng$$

12
 13
 14 Using these mapping, we are able to plot user his-
 15 torical traces onto grid after every δt time interval as
 16 shown in Figure 3. After plotting the user traces on
 17 the grid, we apply statistical mode operation (defined
 18 below) to determine the user location at given time t .

20 **Temporal Mode of User Location Points:**

21 The temporal mode of a set of data points is the
 22 value in the set that occurs most often during a spec-
 23 ified time interval. The mode of i^{th} user's historical
 24 data points, $mode(Tr_i)$, is the value that occur most
 25 frequently within a specified the time. For applying the
 26 mode operation, users longitude and latitude values are
 27 mapped onto the grid so that it points to the defined
 28 geographical region. The mode operation is applied for
 29 each user.

30 Let Tr_i be the set of GPS points or trajectory of
 31 user i at time δt and l_i be the most frequently visited
 32 location point of user for given time δt .

$$33 \quad l_1 = mode(Tr_1)$$

$$34 \quad l_2 = mode(Tr_2)$$

$$35 \quad l_n = mode(Tr_n) \quad (1)$$

$$36 \quad L = \{l_1, l_2, l_3, \dots, l_n\}$$

37 For a set of users, we can determine the cell num-
 38 bers(location points) in which they are expected to be
 39 at any point of time. Let the set of location points of n
 40 users is denoted with $L = \{l_1, l_2, \dots, l_n\}$ at time t .

41 By applying these statistics we determine the cell
 42 where an individual user is mostly present at a given
 43 point of time. We discard the cells which are visited
 44 very few number of times in long period of GPS traces.

45 A baseline algorithm to solve the meeting point prob-
 46 lem is defined below.

47 **4.2 Baseline Algorithm**

48 For a given set of users, let L be the union of set of lo-
 49 cation points. The baseline algorithm considers all the
 50 cells $|C|$ within the grid as the probable candidates for

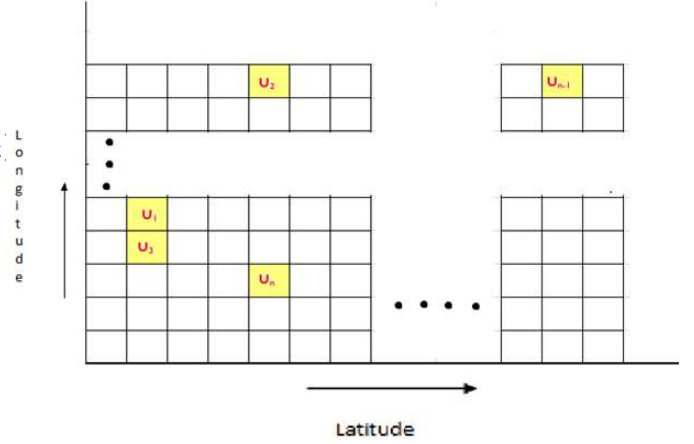


Fig. 3: Location Point Grid

optimal meeting point. The baseline algorithm evalu-
 ates the sum of distances from the location points of
 each user to each cell and the OMP is the cell with the
 minimum value of the sum.

The approach is presented in Algorithm 1 where
 function d_N computes shortest distance between a lo-
 cation point l_i and cell c_j on road network using any
 standard procedure (in our case Google Maps API).

The baseline algorithm has very high computational
 complexity $O(nK)$, where n number of users and K is
 the number of cells, as it considers the entire grid as
 the search space. It is required to prune the space to
 overcome this problem. We propose the use of a two
 level convex hull pruning to reduce the search space,
 and there by improving the efficiency.

Algorithm 3: BaseLine Algorithm(L,C)

Data: Location points $L = \{l_1, l_2, \dots, l_n\}$ of n users at
 given time T_i on the Grid containing $|C|$ cells

Result: Optimal Meeting Place- A cell on the Grid
 $OMP(Lat, Lng)$

```

begin
  OMP ← NULL
  mincost ← +∞
  foreach  $c_i \in C$  do
    sum ← 0
    foreach  $l_j \in L$  do
      sum ← sum +  $d_N(c_i, l_j)$ 
    cost ← sum
    if cost < minCost then
      mincost ← cost
      OMP ←  $c_i$ 
  Return(OMP)

```

So baseline approach is to consider all the cells on the grid as a search space to determine the optimal meeting point. In our approach we use two level convex hull pruning to reduce the search space and to improve the efficiency of search.

4.3 Convex Hull Based Pruning

The convex hull $H(L)$ of a set L is the intersection of all convex sets of which L is a subset. It is also the union of all straight lines joining all pairs of points in L [27].

It can be observed that given a set of location points L , a minimum distance point from all location points of set L , i.e., $\text{argmin}_{x'}[\sum d_E(l_i, x')]$ always lies inside the convex hull $H(L)$, where function $d_E(x, y)$ returns euclidean distance between points x and y . It can be deduced from the property of a convex object that its centroid lies within the object [27]. But, as shown in Figure 4(a), it may not be always true for road network. To ensure this property for road network, we calculate the shortest route between every two location points using function $\text{shortRoute}(x, y)$, all the points that lies among the routes are merged with location points set as described in Figure 4(b). After that we take the convex hull of this set.

According to the baseline algorithm, OMP must exist among one of cells in the grid. It is not necessary to check all the cells in the grid. The search space can further be pruned. We check only those cells that are in the smallest partitioned grid enclosing all cells of the user's location points. We define a convex hull based pruning technique in Algorithm 4, where $\text{convexHullPath}(L)$ computes the convex hull of the point set L using Andrew's Monotone Chain algorithm [28] and takes $O(|P|\log|P|)$ time where P is the number of points. All the cells those lie within the hull, are collected into set P . Now to determine the OMP we check only those that are belong to set P .

The search space is significantly reduced using convex hull based pruning. It is further possible to trace the direction of movement of the user from her GPS traces. We can further reduce the search space by considering the direction of movement of the users'.

4.4 Direction of Movement Based Pruning

Let $L(t) = \{l_1, l_2, \dots, l_n\}$ be the set of location points of users at time t and let P be the set of cells of the grid lying within the convex hull of $L(t)$. Similarly, let $L'(t + \delta t) = \{l'_1, l'_2, \dots, l'_n\}$ be the set of location points at time $t + \delta t$ and P' be the set of cells of the grid lying within the convex hull of $L'(t + \delta t)$.

Algorithm 4: ConvexHullPruning(LocationPoints, Cells)

Data: Location points $L = \{l_1, l_2, \dots, l_n\}$ of n users at given time T_i on the Grid consists of cells C

Result: Set of Grid Cells P lying inside the Convex Hull

```

begin
  OCells ← 0
  foreach  $l_i \in L$  do
    foreach  $l_k \in L$  do
       $OCells \leftarrow \text{shortRoute}(l_i, l_k)$ 
   $L_n \leftarrow L \cup OCells$ 
   $P \leftarrow 0$ 
   $H \leftarrow \text{ConvexHullPath}(L_n)$ 
  foreach  $c_i \in C$  do
    if  $c_i \in H$  then
       $P \leftarrow P \cup c_i$ 
    else
      Discard  $c_i$ 
  Return( $P$ )

```

As shown in the Figure 5(a), convex hull formed by five different users locations (labeled as 1, 2, 3, 4 and 5) before meeting marked by green lines intersect with after meeting locations marked by red line. In the Figure 5(b) two convex hulls are totally disjoint. These two consecutive convex hulls may be overlapped, contained, intersecting or disjoint. But all these possibilities are mainly categorized into two cases.

Case 1: The two convex hulls are intersect or overlap.

In this case, the meeting point among the cells is assumed to lie inside the intersection/overlapped region. Figure 6 depicts the intersecting and Figure 7 depicts the overlapped convex hulls of four users. It may be noted that this case also covers the case if a convex hull is completely contained in the other convex hull shown in Figure 8. In this case, we assume that the meeting point would lie in the smaller convex hull. We take meeting point inside the intersection because it reduces the sum of total distance traveled by users before and after the meeting.

Case 2: The two convex hulls are disjoint of each other (i.e. they have zero intersection). In this case, we assume that the meeting point would lie in the first convex hull.

These two levels of convex hull pruning prune the search space to P'' , a considerably reduced set. A complete Algorithm 6 is developed to determine optimal meeting point in such a scenario. Function Map(OMP, OMPLoc) converts cell designated as OMP into (Latitude, Longitude) pair represented by OMPLoc.

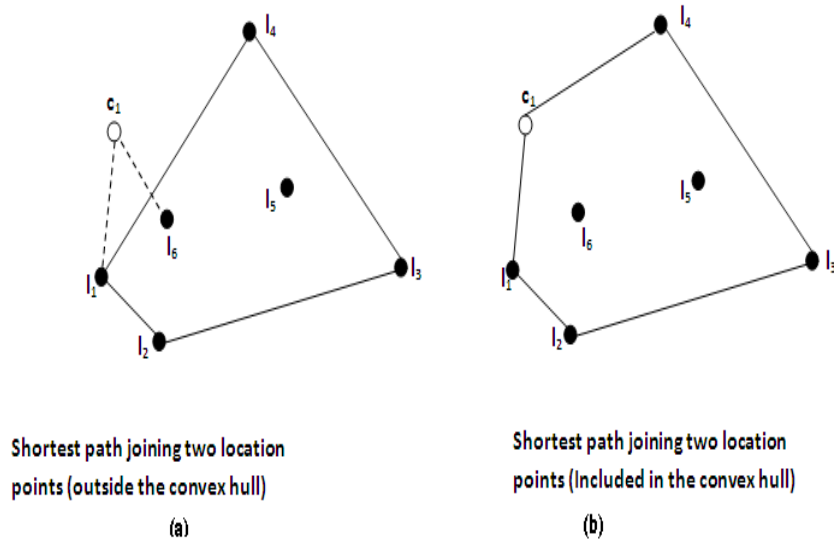


Fig. 4: Counter example

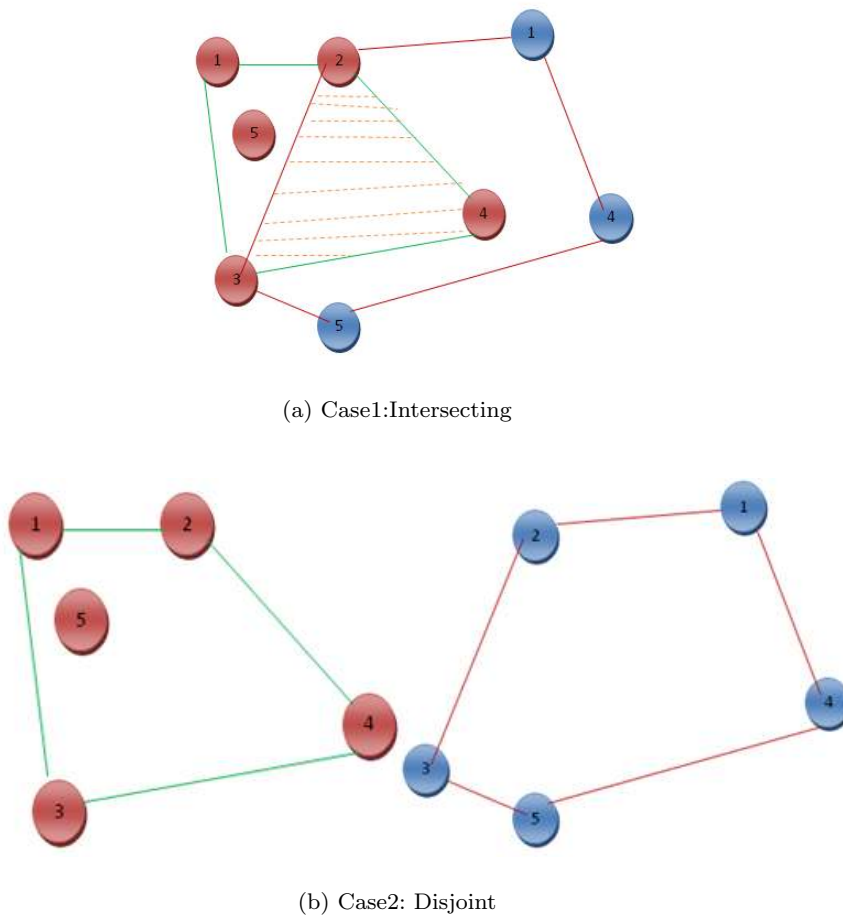


Fig. 5: Example cases of Two Levels of Convex Hulls

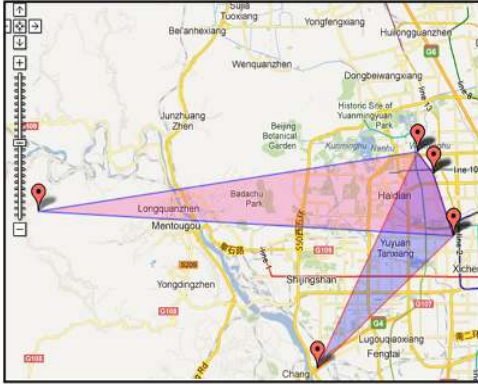


Fig. 6: Intersecting Hulls

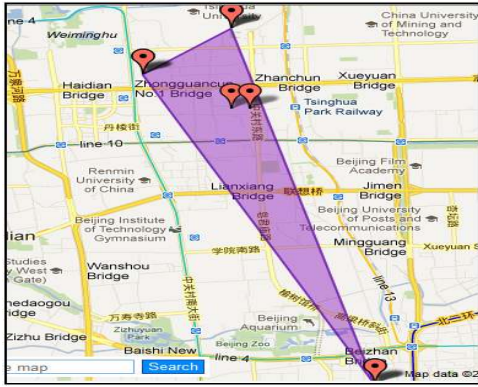


Fig. 7: Overlapped Hulls

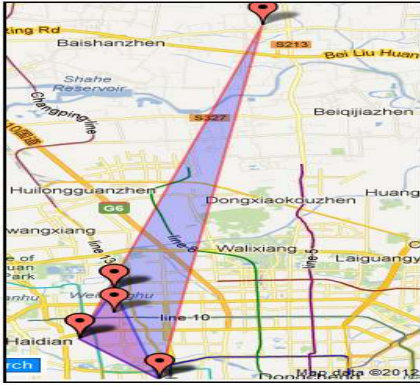


Fig. 8: A hull inside another hull

5 Data Analysis and Results for Variable Number of Users

In this section, we first present details about the GPS dataset used [2,3,6]. Next we present the results of applying statistical operations on the temporal aspect of GPS data for predicting user location points at a given time. Then, two levels of pruning are used to determine OMP.

Algorithm 5: DirectionPruning(LocationPoints,Cells)

Data: Set of Grid Cells P at given time T_i and P' at time $T_i + \delta t$

Result: Set of Grid Cells P'' selected after pruning

```

begin
   $P'' \leftarrow P \cap P'$ 
  if  $P''$  is Null then
     $P'' \leftarrow P$ 
  Return( $P''$ )

```

Algorithm 6: OMP Algorithm(L, L', C)

Data: Location points $L = \{l_1, l_2, \dots, l_n\}$ and $L' = \{l'_1, l'_2, \dots, l'_n\}$ of n users at given time T_i and $T_i + \delta T_i$ on the Grid G containing $|C|$ cells

Result: Optimal Meeting Place- A cell on the Grid $OMPLoc(Lat, Lng)$

```

begin
   $P \leftarrow convexHullPruning(L, C)$ 
   $P' \leftarrow convexHullPruning(L', C)$ 
   $P'' \leftarrow DirectionPruning(P, P')$ 
   $mincost \leftarrow +\infty$ 
  foreach  $c_i \in P''$  do
     $sum \leftarrow 0$ 
    foreach  $l_j \in P''$  do
       $sum \leftarrow sum + d_N(c_i, l_j)$ 
     $cost \leftarrow sum$ 
    if  $cost < minCost$  then
       $mincost \leftarrow cost$ 
       $OMP \leftarrow c_i$ 
  Map( $OMP, OMPLoc$ )
  Return( $OMPLoc$ )

```

We compare the results of our approach with two other approaches in terms of total distance traveled by users before and after the meeting and number of cells searched. The first approach is when we do not consider the direction of movement of users and second approach is when we consider all location points before and after the meeting i.e. L and L' simultaneously. Finally, we compare our approach with related study.

5.1 GPS Trajectory Dataset

We worked on 126 users from the dataset [2] and worked on their GPS traces. This subset consists of a total of 68612 days data with 5,832,020 GPS points. We have partitioned the total area covered by users into grid. Following subsections explain the grid formation for further processing.

5.2 Determining the Grid

The major portion of this dataset belongs to Beijing, China. A grid is defined over this area, which is approximately 38972.068 Sq. kilometers, with minLat=38.0, minLng=115.0 and maxLat=40.0, maxLng=117.0. This is further divided into small cells of (222 meter * 219 meter) area with $\delta\text{Lat}=0.002$ and $\delta\text{Lng}=0.025$. Thus we have a total of 1,800,000 cells. These cells are sequenced major rowwise, minor columnwise and identified by unique numbers from 1 to 1800K. Users GPS points are mapped into these cells and their location points are predicted.

After partitioning the city into the grid, the aim is to compute the user locations at a given point of time and map them onto grid.

5.3 Predicting User Location Points

Recent one month historical data of a user is analyzed to predict his/her location at a given time. For example if we have to predict the user location for 18 April at 10 am. then we extract his last month data points for the time interval 9:45 to 10:15 am. Data points are mapped on to grid and temporal mode operation is applied to determine the cell with maximum frequency of data points. This cell is marked as location point for the user at 10 am.

The location is evaluated for hundreds times and prediction is made for days for which data is already available. It is observed that 73% of times predicted cell is same as actual location of user. For all predictions made, a **mean square error of 0.238** was determined. Figure 9 shows a mean square error detected for the 104 predictions made.

After computing users locations our aim is to determine a meeting point for them on the road network.

5.4 Determining Optimal Meeting Place

After computing users location points, we determine a meeting point for them on road network. For this we apply two levels of convex hull pruning algorithms (Algorithm 4 and Algorithm 5). Some of the results and different consecutive convex hull cases like intersecting, contained and overlapped are shown in Figure 10 and Figure 11. To check the efficiency and accuracy of our approach, we determined optimal meeting point in three different ways, described below.

5.4.1 OMP by considering set L (No Directions)

In this case, we determined the meeting point, for a given meeting time, by considering only set L . For example, if the meeting time is at 10 am, then search space is pruned by considering users' location at 10 am only. A convex hull of current location points L is calculated and the optimal meeting point is determined within this convex hull.

5.4.2 OMP by considering set L and L' simultaneously

Here we determine the meeting point, for a given meeting time, by considering both set L and L' at the same time. For example, if the meeting time is at 10 am and meeting duration is one hour, then we prune the search space by using both sets of user locations at 10 am and at 11 am simultaneously. Convex hull of all location points is calculated and the optimal meeting point is determined within this convex hull.

5.4.3 OMP by considering Direction of Movement of Users with Time

In this case, we consider the direction of movement of users to determine the meeting point. For example, if meeting time is 10 am and meeting duration is one hour, then we prune the search space by considering users' location points at 10 am and at 11 am. We first determine the convex hull of locations points at 10 am and then we determine the convex hull of user locations at 11 am. An optimal meeting point is determined within the intersecting/overlapped area of two convex hull.

A Table 2 summarizes the average distance traveled by all users and average number of cells to be searched in each of these three cases is given below.

Calculations are performed 50 times for different number of users and optimal meeting point is determined in each of these three cases. Figure 12 shows the graph of total distance traveled by users in each of these three cases. The X-axis shows the average distance traveled by users. Graph shows that users have to travel much more distance in case 1, when we do not consider direction of movement of users. In case 2 and case 3 these distances are nearly same.

Figure 13 shows the graph of number of cells to be searched to determine the optimal meeting point in each of these three cases. Graph shows that more number of cells are required to be search in Case 1 and Case 2 as compare to Case 3. It shows that the search space is significantly reduced if we apply the two-level convex hull pruning.

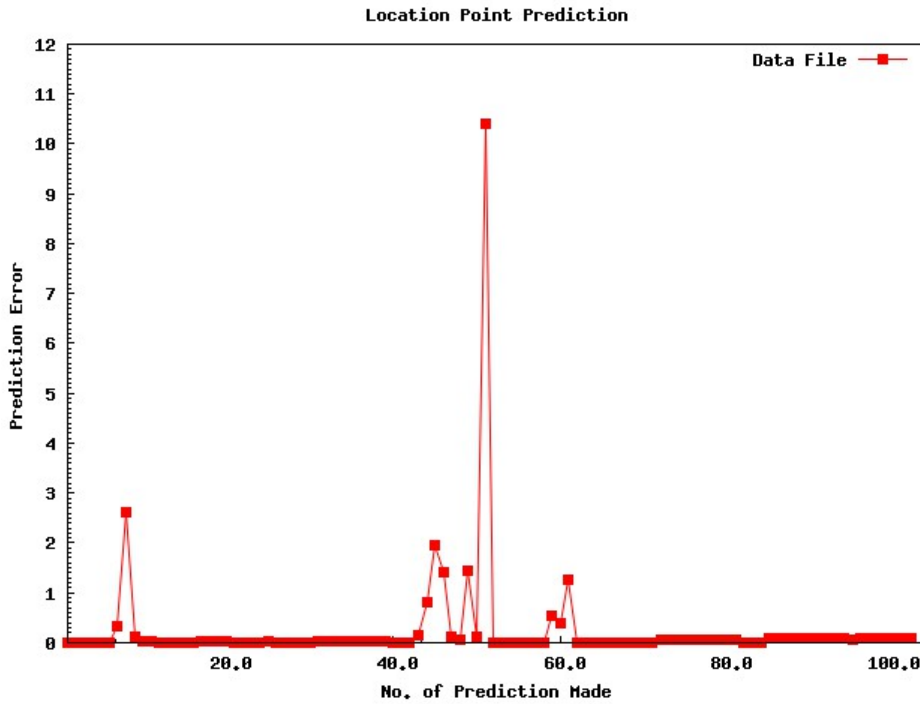


Fig. 9: Prediction Error

Table 2: Distance Travel and Cells to be searched in 3 different Cases

Without Directions		All Location Points		With Directions	
<i>Avg.Distance</i> ₁	<i>Avg.Cells</i> ₁	<i>Avg.Distance</i> ₂	<i>Avg.Cells</i> ₂	<i>Avg.Distance</i> ₃	<i>Avg.Cells</i> ₃
35.032	454.925	31.00505	407.375	31.94225	74.125

6 Comparison with Related Studies

As discussed in the related work, our study falls into the group level OMP determination on the road network. To the best of our knowledge, group-level meeting location discovery studies were conducted by Yan et al. [1].

We compared our study with the study conducted by Yan et al. (2011) based on two criteria; total distance required to travel by the users before and after the meeting and the number of cells required to be searched to determine the optimal meeting point.

Yan et al. [1] proposed a convex hull based method to determine OMP. In their approach, they considered all the query points fixed, but in our approach, we considered moving query points (users). In real life also query points are never stationary.

Yan et al. [1] approach, they reduced the total search space by taking the convex hull of all stationary query points, on the other hand, we reduced it further by considering the users' direction of movement after the meeting. By applying the two levels of convex hull based pruning search space get significantly reduced.

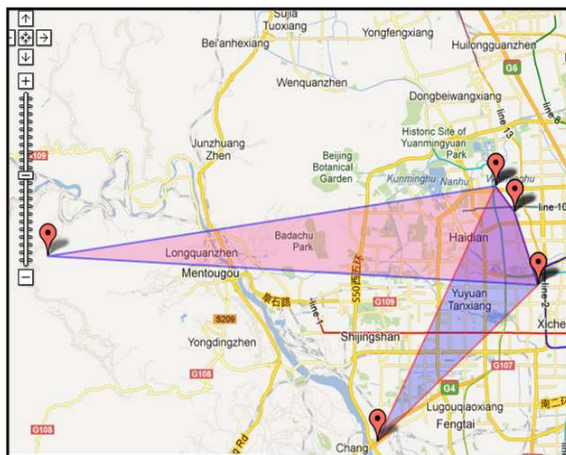
In Figure 12 first bars (red bars) in the bar-graph gives the total distance travelled by the users considering users' locations as stationary and third bars (blue bars) show the value when users are moving (our approach). It is determined that the average distance travelled by users is 34km in our case and it is 39km when users are stationary.

Similarly, in Figure 13 first bars (red bars) in the bar-graph indicates the total number of cells are required to be searched when considering users' locations as stationary and third bars (blue bars) show the value when users are moving (our approach). It is determined that the average number of cells required to be searched in our case is approximately 78 whereas it is approximately 626 cells when users are stationary.

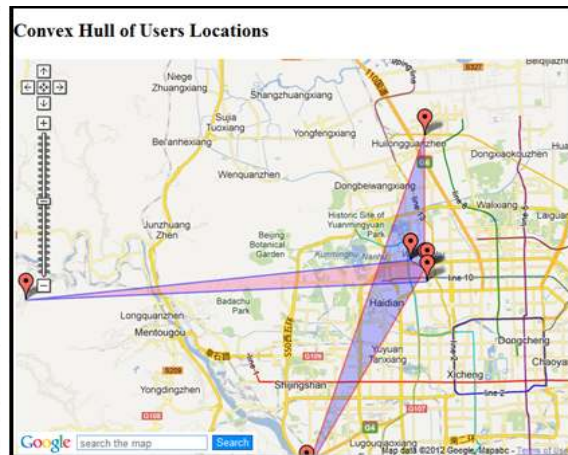
7 Conclusions

In this paper, we proposed the solution to determine Optimal meeting for the parties who are situated in geographically distant locations of a city and have varying traveling patterns.

We proposed a model to determine the meeting point for the above problem for groups of two users from their



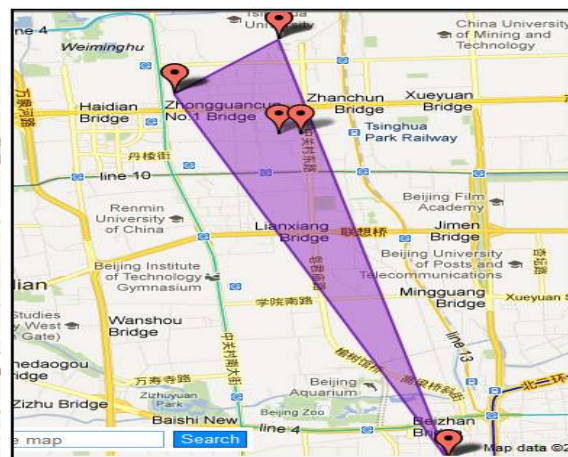
(a) Intersecting



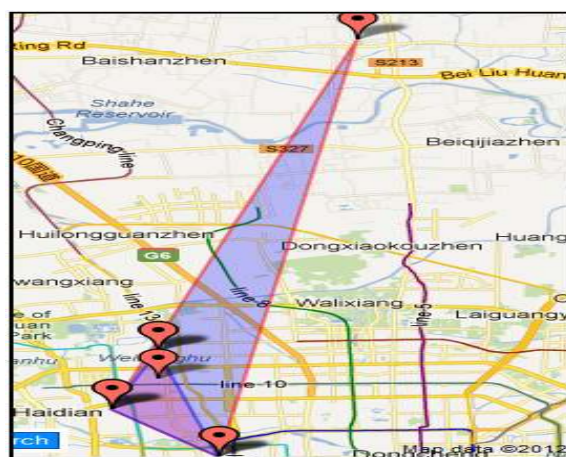
(b) Intersecting



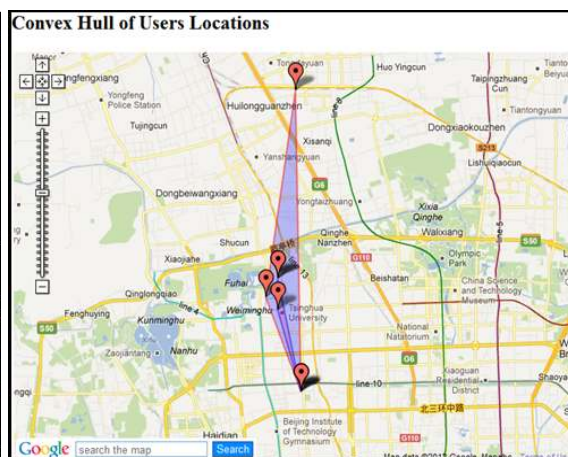
(c) Intersecting



(d) Overlapped



(e) Contained



(f) Contained

Fig. 10: Sample Results of Two Levels of Convex Hulls

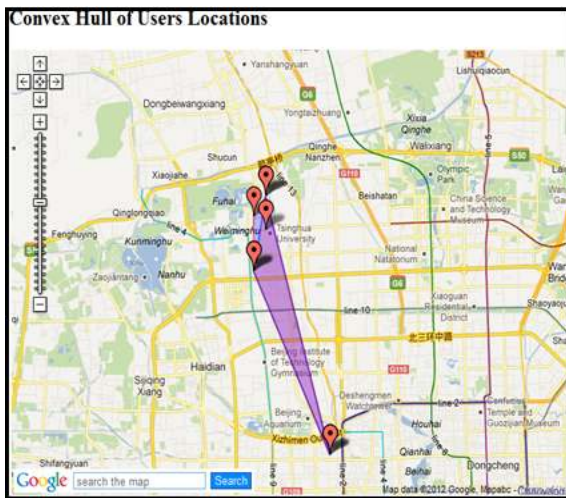


Fig. 11: Contained

GPS traces. We begin by mining stay points and interesting locations within a given geo-spatial region. We then constructed the travel graph of each user. Using the graphs of two users it is possible to effectively arrange meetings between them.

Then, we investigated the problem of identifying a common meeting point for a group of users who have temporal and spatial locality constraints that vary over time. We modeled the above problem for a number of moving users on road network by using the GPS traces of the users. We applied two levels of convex hull based pruning and determine the optimal meeting point. We have used predicted future direction of movement of the users to reduce total distance traveled by users before and after the meeting.

The method was evaluated on a large real-world GPS trace dataset and showed the effectiveness of our proposed method in identifying a common meeting point for an arbitrary number of users on the road network.

References

- Da Yan, Zhou Zhao, and Wilfred Ng. Efficient algorithms for finding optimal meeting point on road networks. *Proceedings of the VLDB Endowment*, 4(11), 2011.
- Yu Zheng, Longhao Wang, Ruochi Zhang, Xing Xie, and Wei-Ying Ma. Geolife: Managing and understanding your past life over maps. In *The Ninth International Conference on Mobile Data Management (mdm 2008)*, pages 211–212. IEEE, 2008.
- Yu Zheng, Quannan Li, Yukun Chen, Xing Xie, and Wei-Ying Ma. Understanding mobility based on gps data. In *Proceedings of the 10th international conference on Ubiquitous computing*, pages 312–321. ACM, 2008.
- Da Yan, Zhou Zhao, and Wilfred Ng. Efficient processing of optimal meeting point queries in euclidean space and road networks. *Knowledge and Information Systems*, 42(2):319–351, 2015.
- Sonia Khetarpaul, Rashmi Chauhan, SK Gupta, L Venkata Subramaniam, and Ullas Nambiar. Mining gps data to determine interesting locations. In *Proceedings of the 8th International Workshop on Information Integration on the Web: in conjunction with WWW 2011*, page 8. ACM, 2011.
- Yu Zheng, Lizhu Zhang, Xing Xie, and Wei-Ying Ma. Mining correlation between locations using human location history. In *Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems*, pages 472–475. ACM, 2009.
- X. Xie Y. Zheng, L. Zhang and W.-Y. Ma. Understanding mobility based on gps data. In *In Proceedings of Ubicomp*, pages 312–321. ACM Press, September 2008.
- Tzvetan Horozov, Nitya Narasimhan, and Venu Vasudevan. Using location for personalized poi recommendations in mobile environments. In *International Symposium on Applications and the Internet (SAINT'06)*, pages 6–pp. IEEE, 2006.
- Hoyoung Jeung, Qing Liu, Heng Tao Shen, and Xiaofang Zhou. A hybrid prediction model for moving objects. In *2008 IEEE 24th International Conference on Data Engineering*, pages 70–79. Ieee, 2008.
- Rainer Simon and Peter Fröhlich. A mobile application framework for the geospatial web. In *Proceedings of the 16th international conference on World Wide Web*, pages 381–390. ACM, 2007.
- Ashweeni Beeharee and Anthony Steed. Exploiting real world knowledge in ubiquitous applications. *Personal and Ubiquitous Computing*, 11(6):429–437, 2007.
- Moon-Hee Park, Jin-Hyuk Hong, and Sung-Bae Cho. Location-based recommendation system using bayesian users preference model in mobile devices. In *International Conference on Ubiquitous Intelligence and Computing*, pages 1130–1139. Springer, 2007.
- Leon Cooper. An extension of the generalized weber problem. *Journal of Regional Science*, 8(2):181–197, 1968.
- Lawrence M Ostresh. The multifacility location problem: Applications and descent theorems. *Journal of Regional Science*, 17(3):409–419, 1977.
- Reuven Chen. Location problems with costs being sums of powers of euclidean distances. *Computers & operations research*, 11(3):285–294, 1984.
- Reuven Chen. Solution of location problems with radial cost functions. *Computers & Mathematics with Applications*, 10(1):87–94, 1984.
- Amir Beck and Marc Teboulle. Gradient-based algorithms with applications to signal recovery. *Convex optimization in signal processing and communications*, pages 42–88, 2009.
- Jack Brimberg and Robert F Love. Global convergence of a generalized iterative procedure for the minisum location problem with lp distances. *Operations Research*, 41(6):1153–1163, 1993.
- Kyriakos Mouratidis, Man Lung Yiu, Dimitris Papadias, and Nikos Mamoulis. Continuous nearest neighbor monitoring in road networks. In *Proceedings of the 32nd international conference on Very large data bases*, pages 43–54. VLDB Endowment, 2006.
- Man Lung Yiu, Nikos Mamoulis, and Dimitris Papadias. Aggregate nearest neighbor queries in road networks. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):820–833, 2005.
- Feifei Li, Bin Yao, and Piyush Kumar. Group enclosing queries. *IEEE Transactions on Knowledge and Data Engineering*, 23(10):1526–1540, 2011.

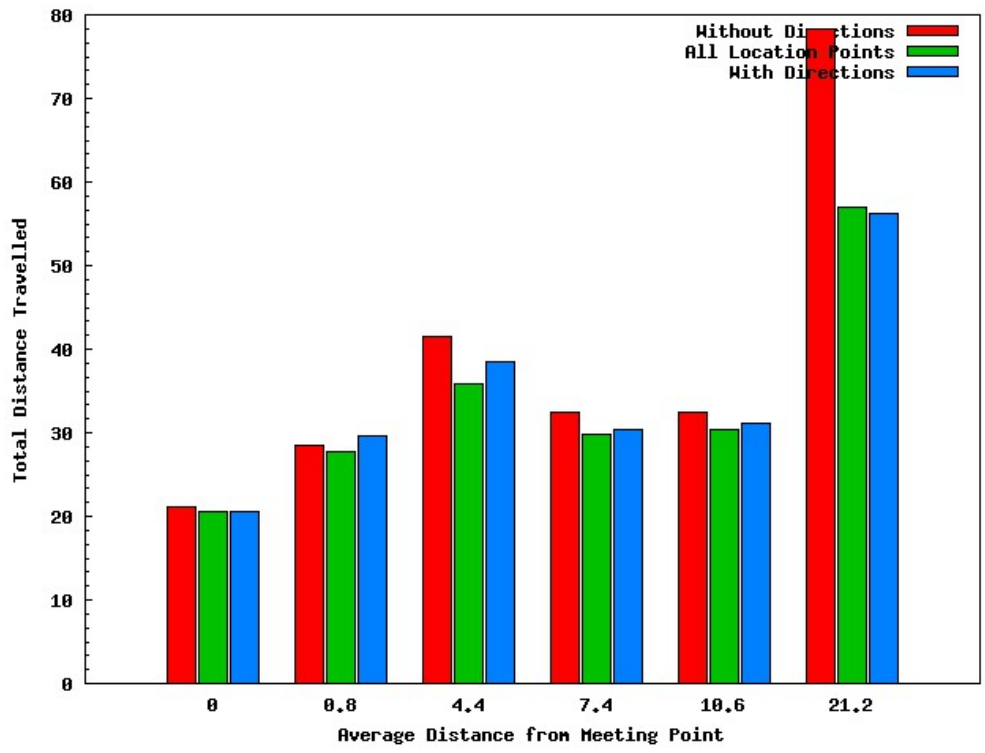


Fig. 12: Distance Traveled by users

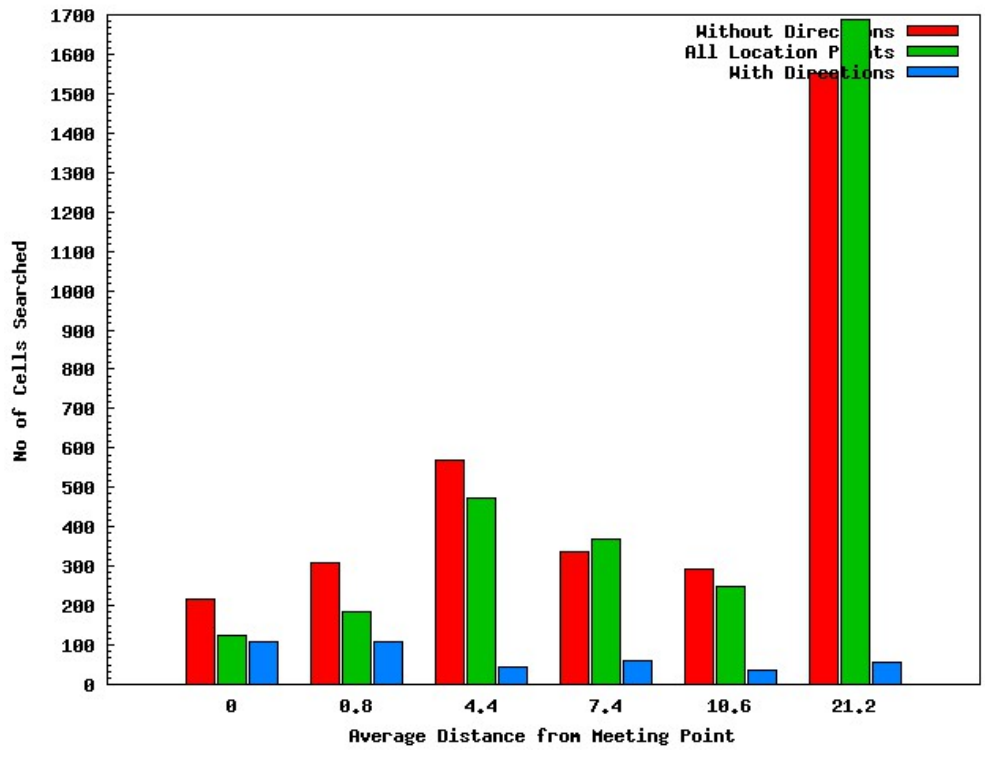
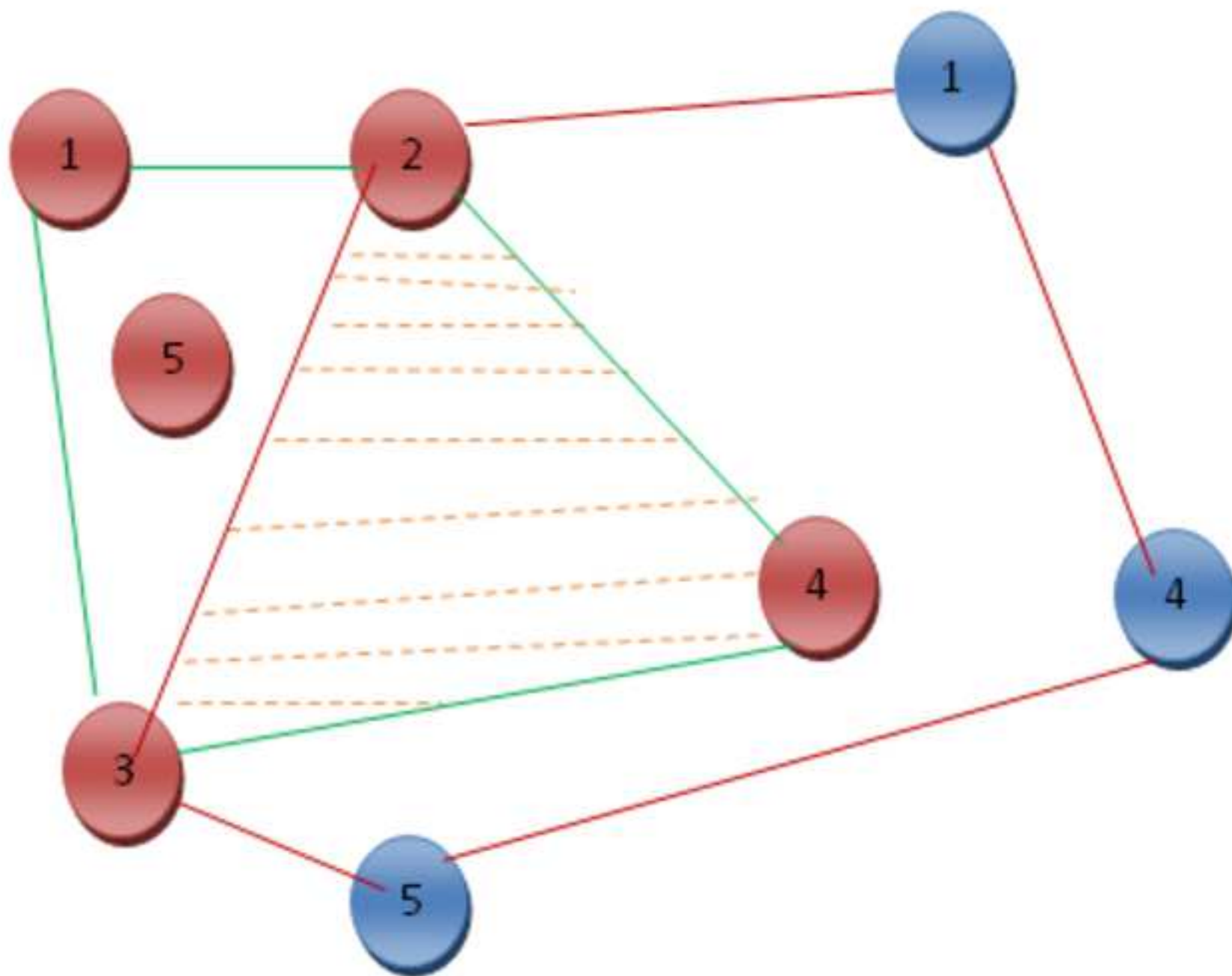


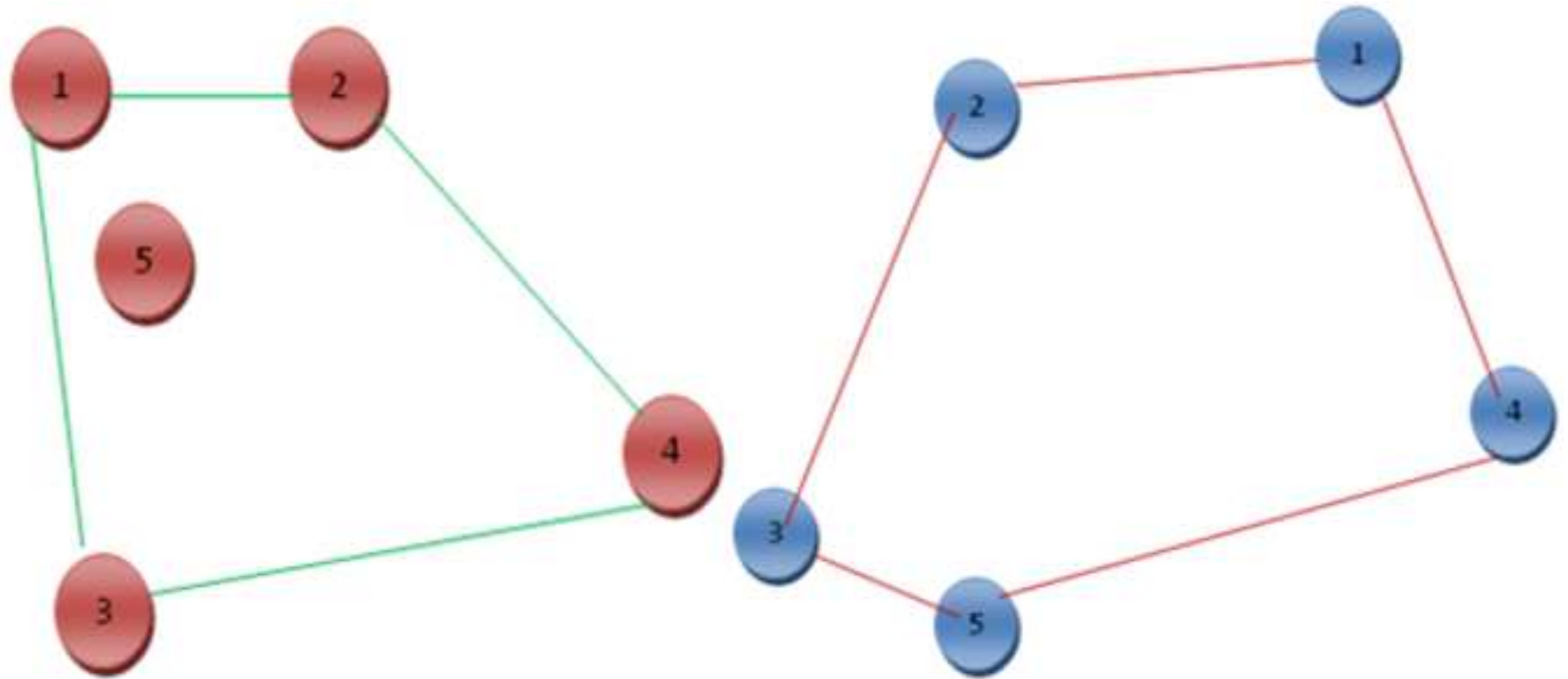
Fig. 13: Search Space reduction

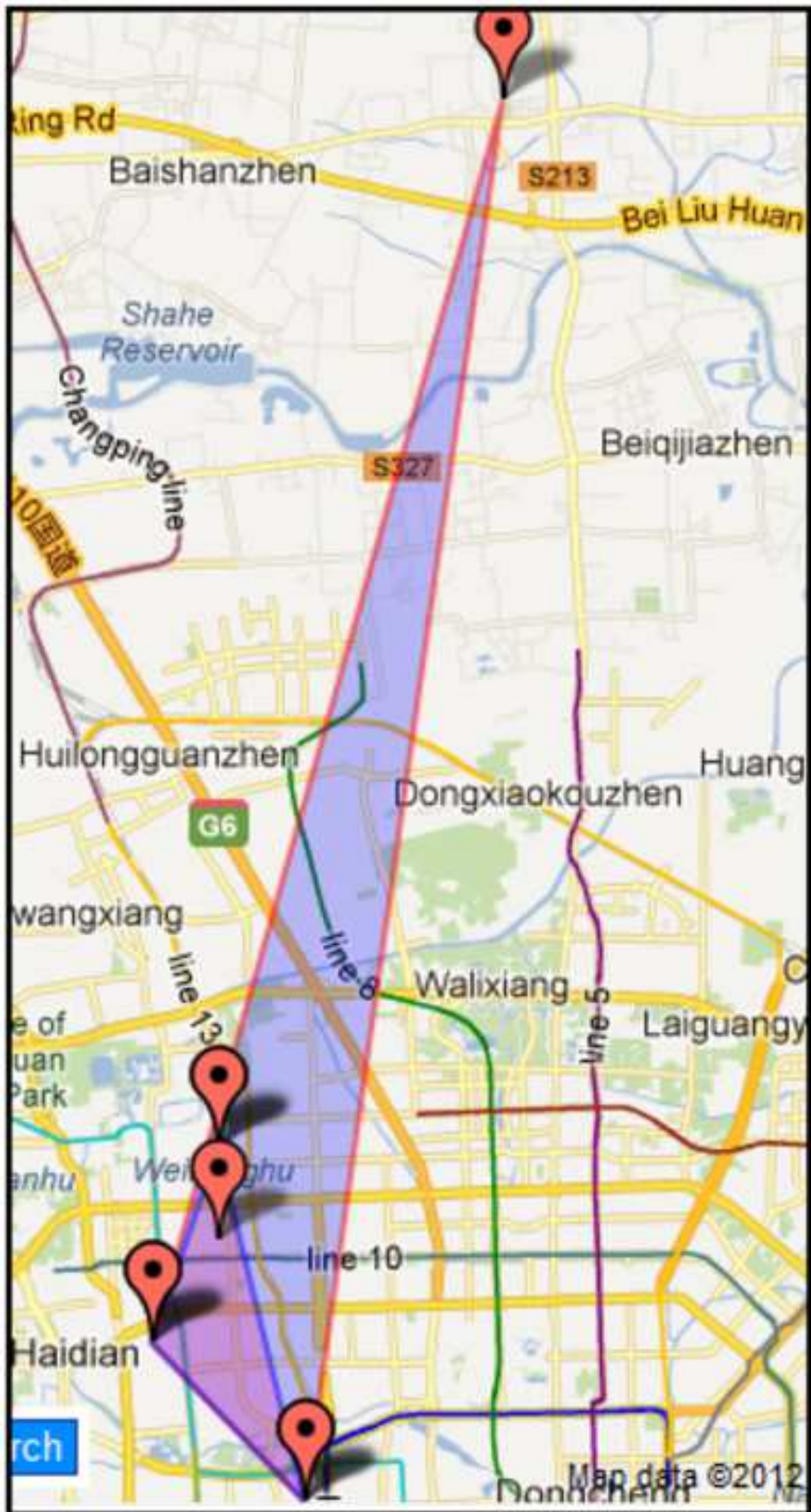
22. Hyung-Ju Cho and Chin-Wan Chung. An efficient and scalable approach to cnn queries in a road network.

In *Proceedings of the 31st international conference on*

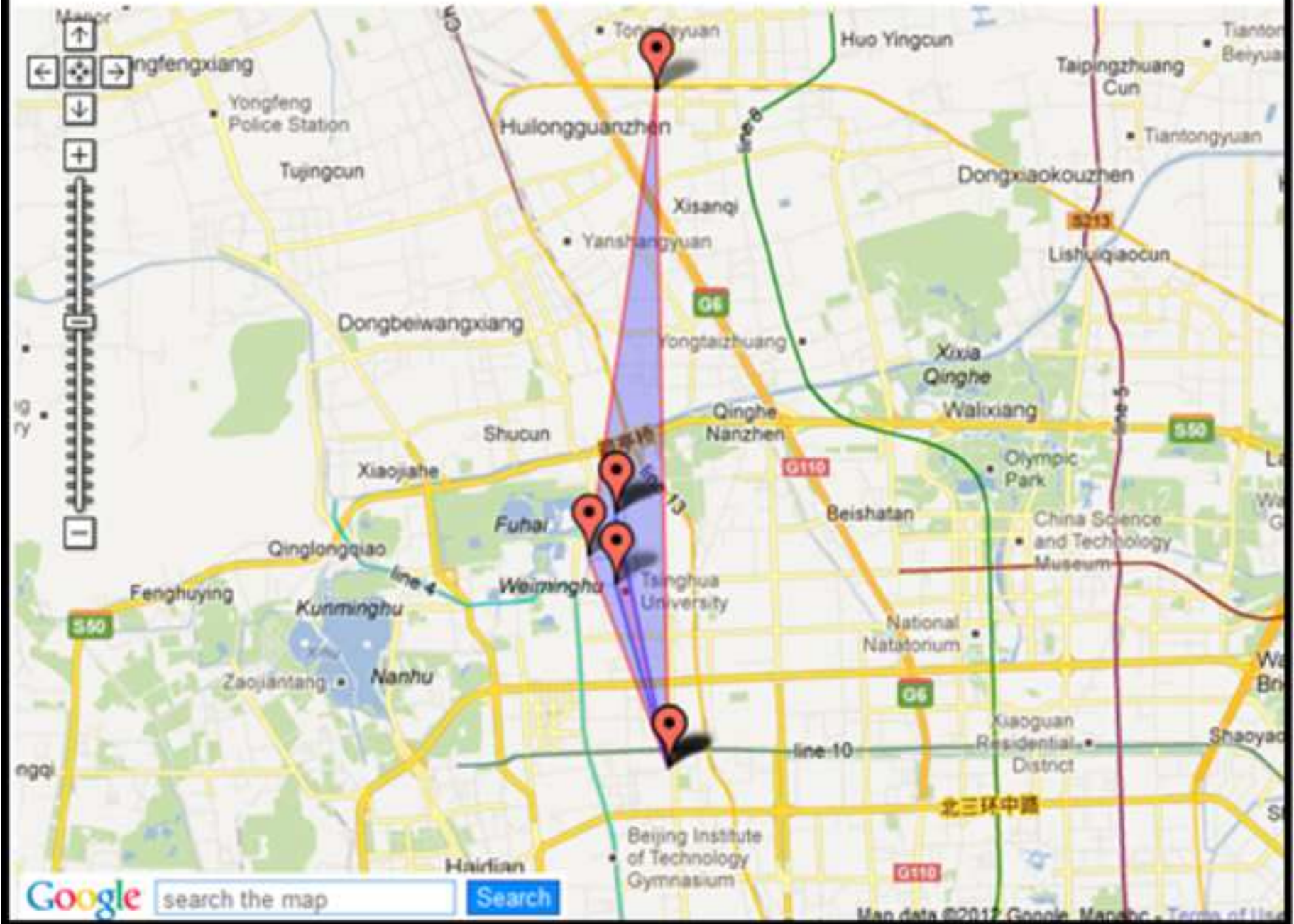
- 1 *Very large data bases*, pages 865–876. VLDB Endow-
2 ment, 2005.
- 3 23. Zaiben Chen, Heng Tao Shen, Xiaofang Zhou, and Jef-
4 frey Xu Yu. Monitoring path nearest neighbor in road
5 networks. In *Proceedings of the 2009 ACM SIGMOD*
6 *International Conference on Management of data*, pages
7 591–602. ACM, 2009.
- 8 24. Zhengdao Xu and Hans-Arno Jacobsen. Processing prox-
9 imity relations in road networks. In *Proceedings of the*
10 *2010 ACM SIGMOD international conference on man-*
11 *agement of data*, pages 243–254. ACM, 2010.
- 12 25. Sonia Khetarpaul, SK Gupta, L Venkat Subramaniam,
13 and Ullas Nambiar. Mining gps traces to recommend
14 common meeting points. In *Proceedings of the 16th In-*
15 *ternational Database Engineering & Applications Sysm-*
16 *posium*, pages 181–186. ACM, 2012.
- 17 26. Sonia Khetarpaul, SK Gupta, and L Venkata Subra-
18 maniam. Analyzing travel patterns for scheduling in
19 a dynamic environment. In *International Conference*
20 *on Availability, Reliability, and Security*, pages 304–318.
21 Springer, 2013.
- 22 27. Convex Hull and properties link.
23 http://en.wikipedia.org/wiki/Convex_set,
24 <http://en.wikipedia.org/wiki/Centroid>. Accessed:
25 2016-01-14.
- 26 28. Franco P Preparata and Michael Shamos. *Computational*
27 *geometry: an introduction*. Springer Science & Business
28 Media, 2012.
- 29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65



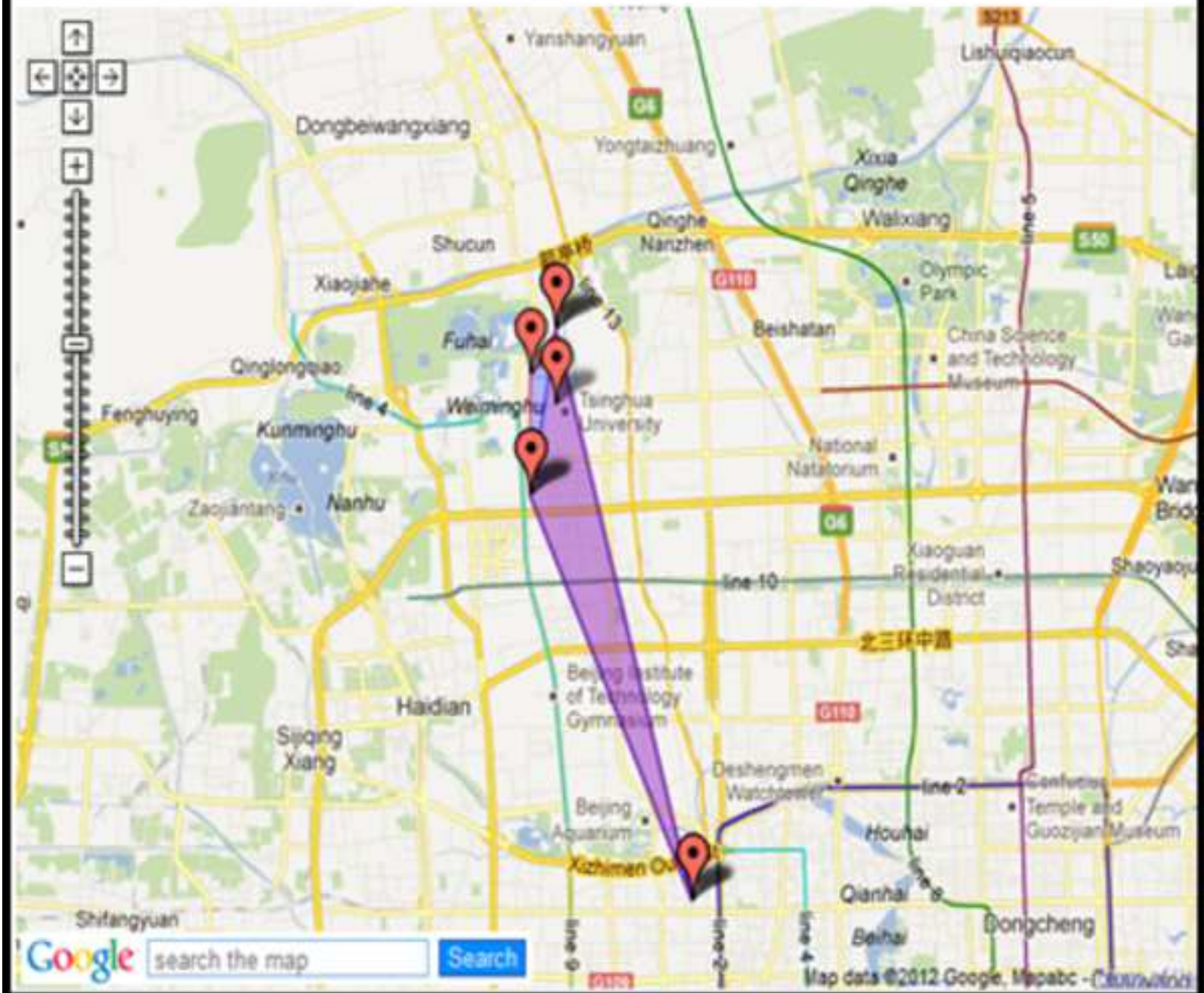


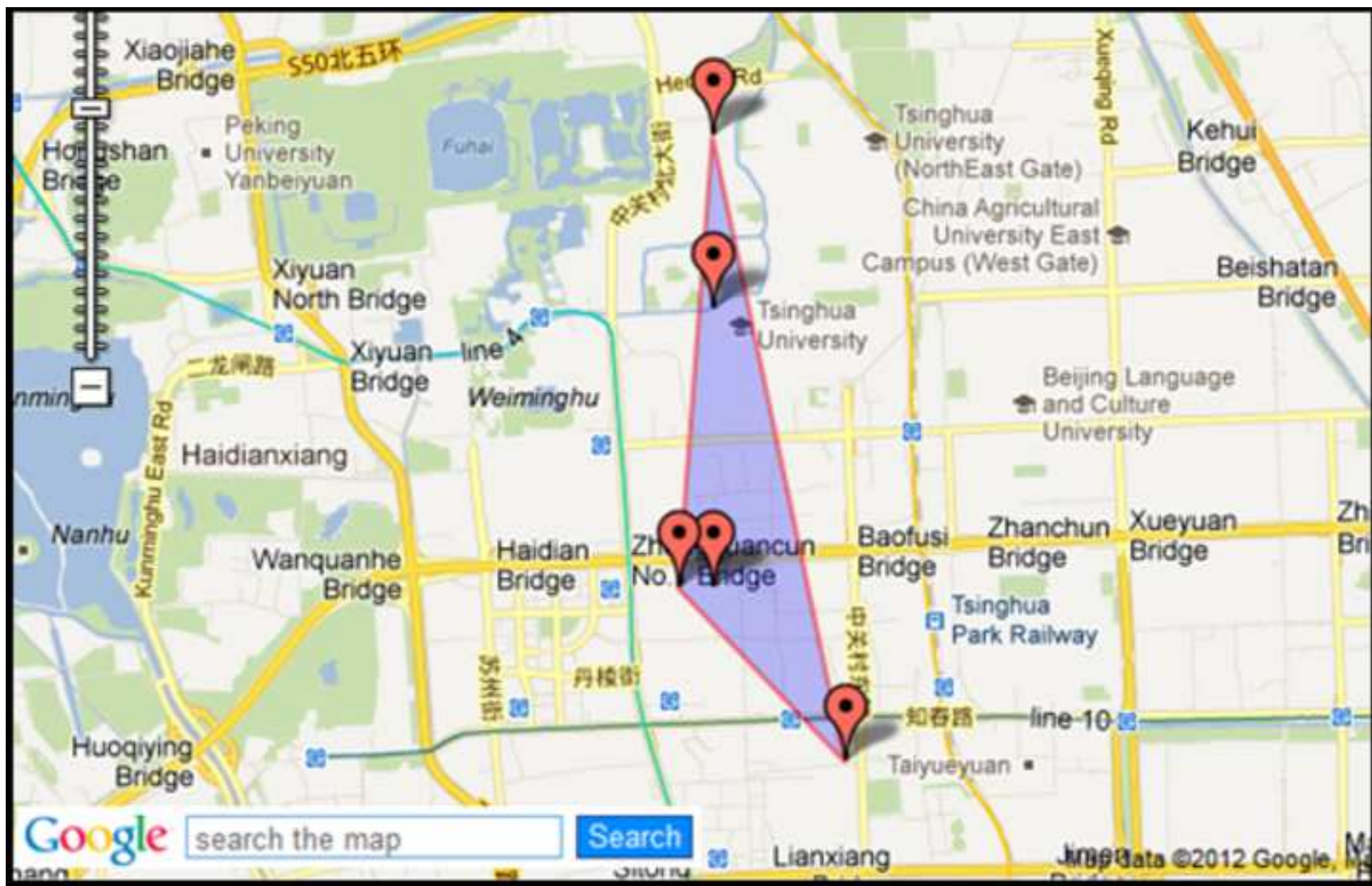


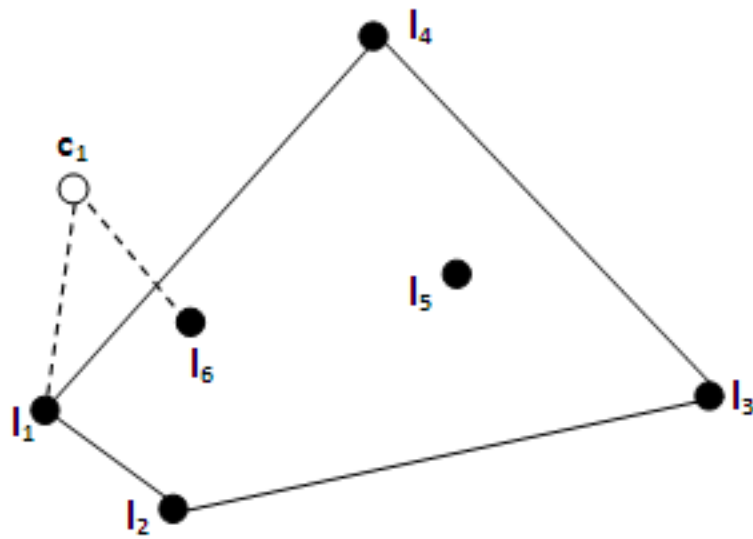
Convex Hull of Users Locations



Convex Hull of Users Locations

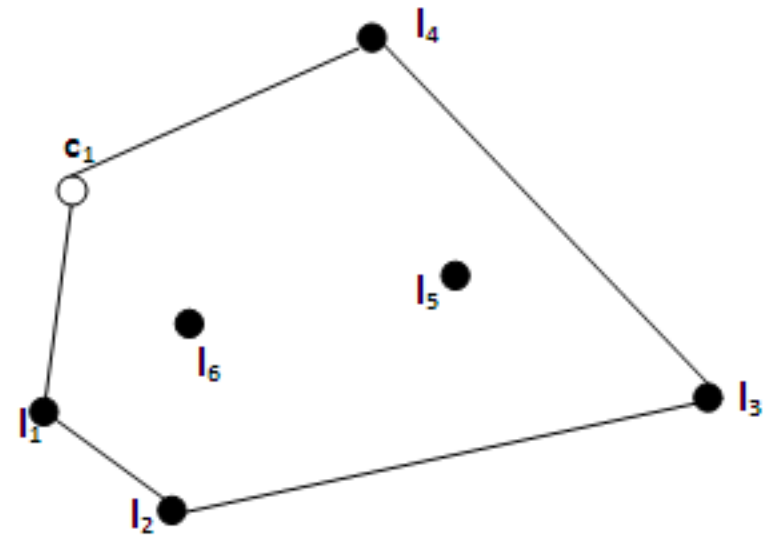






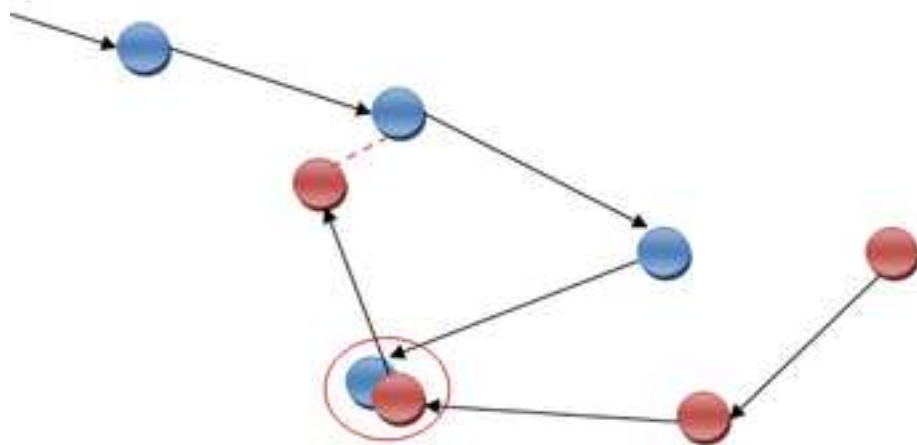
Shortest path joining two location points (outside the convex hull)

(a)

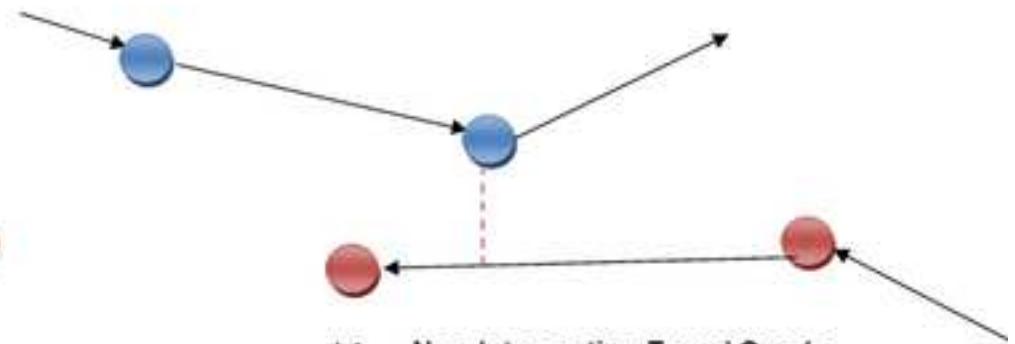


Shortest path joining two location points (Included in the convex hull)

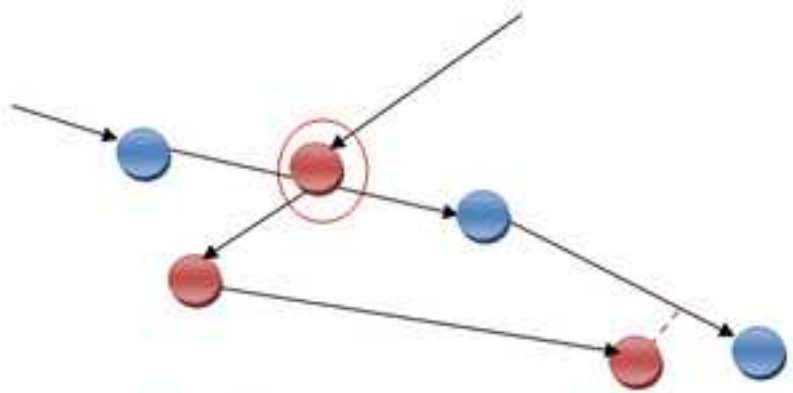
(b)



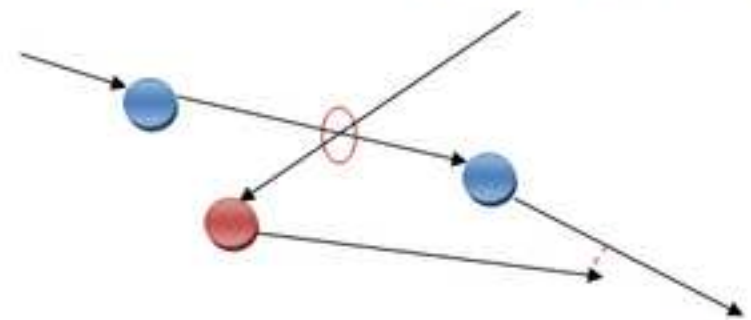
(a) Intersecting Stay points



(c) Non-Intersecting Travel Graphs

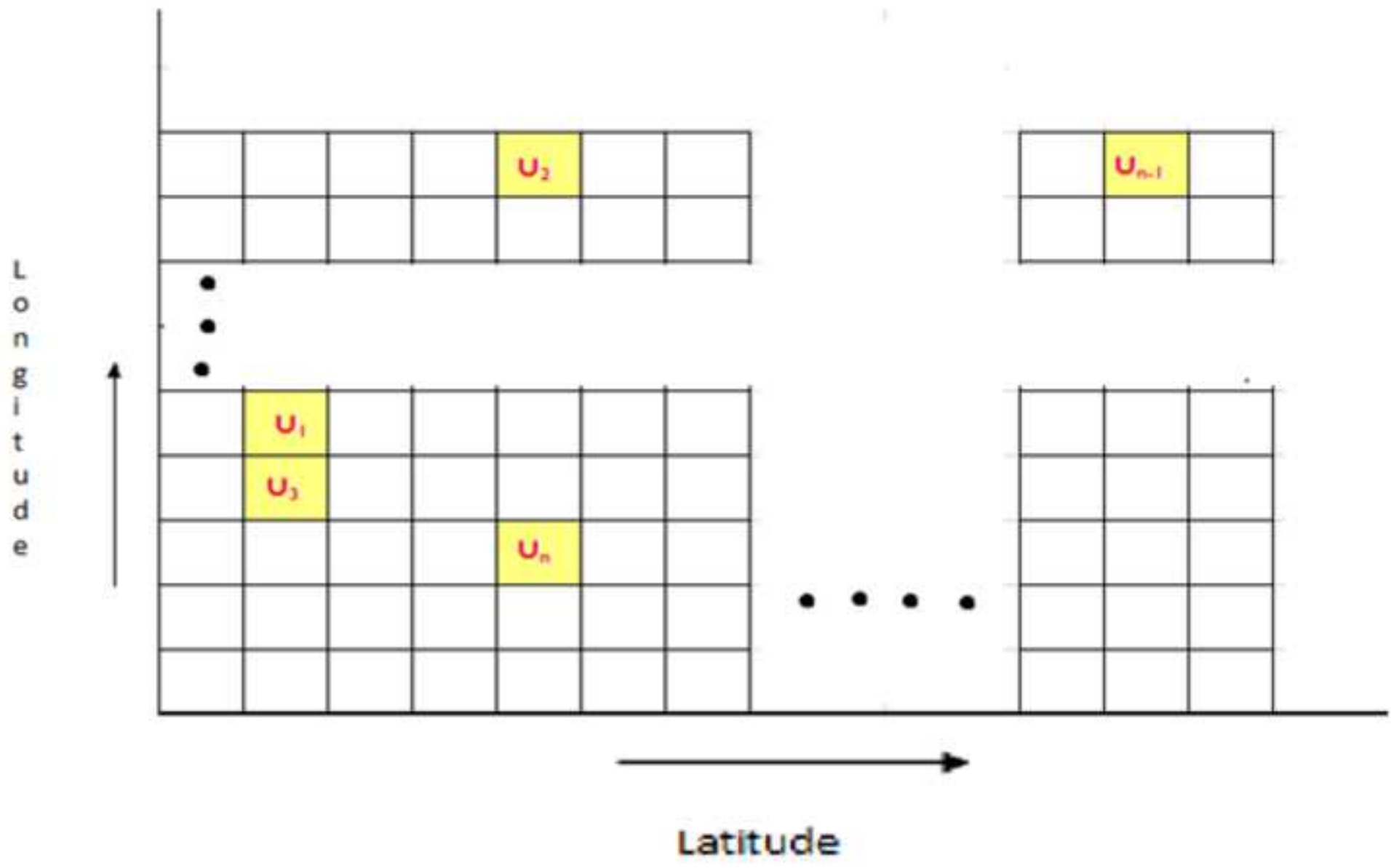


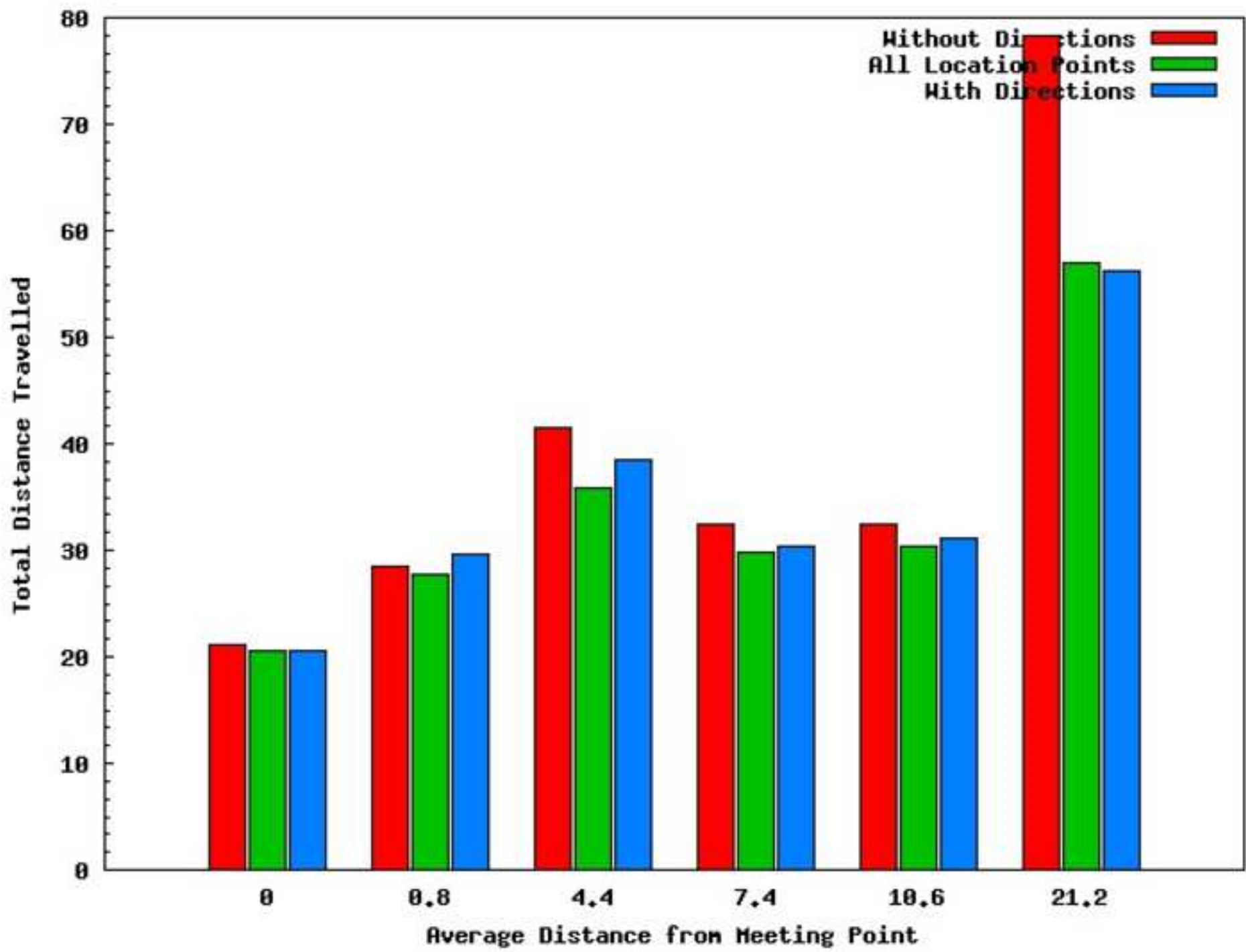
(b) U_i travel path passing through U_j 's

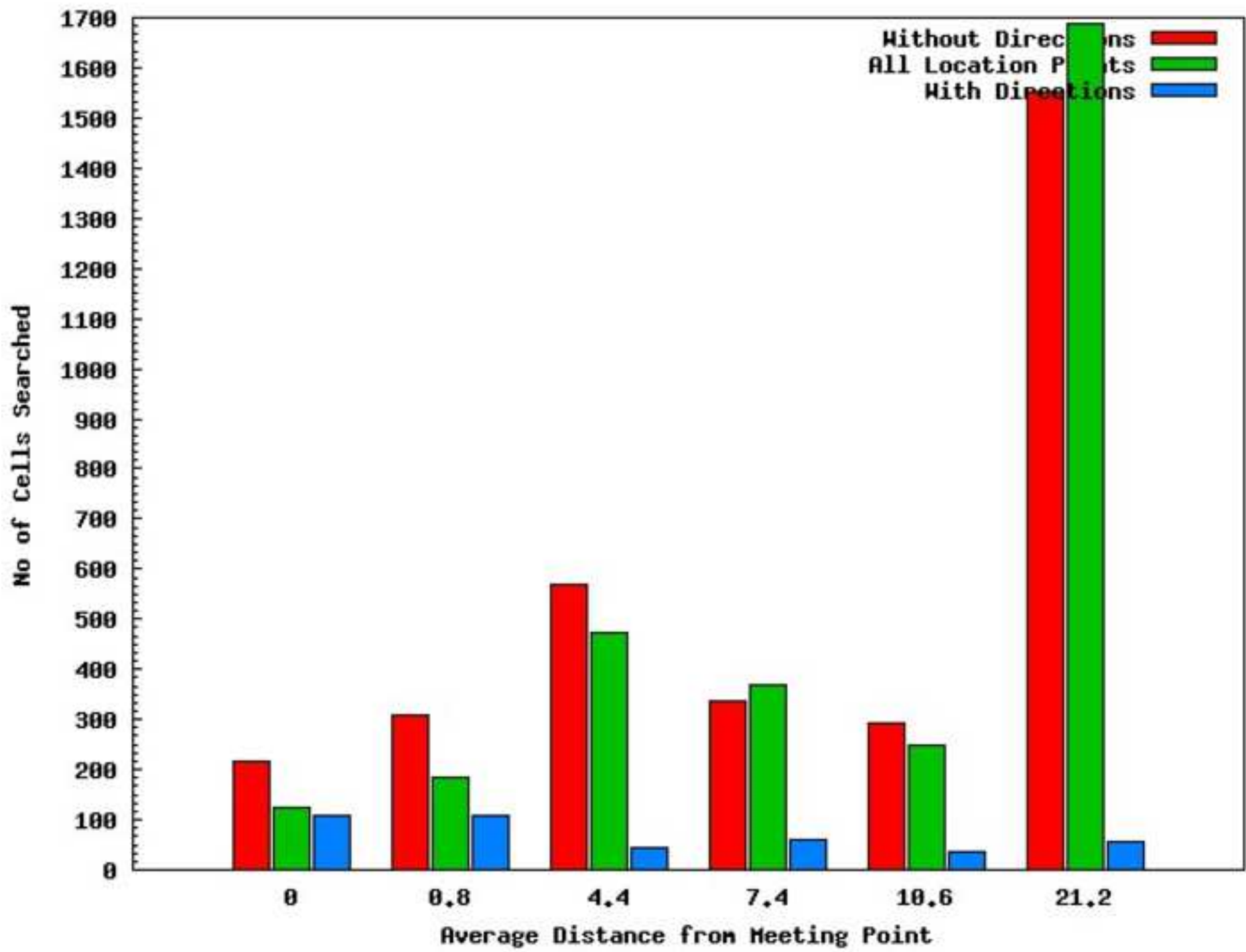


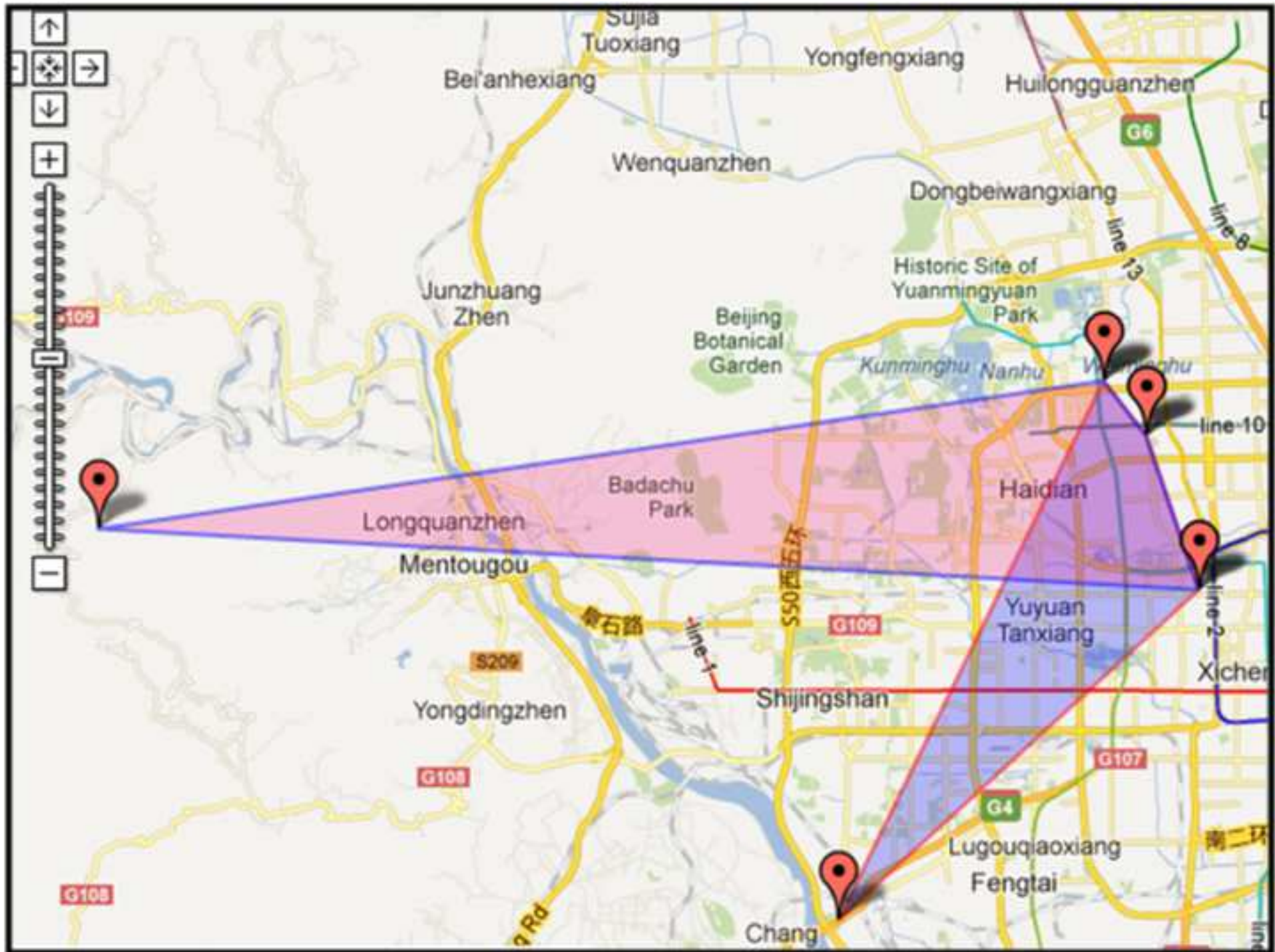
(d) Intersecting Travel Paths

	User _i Stay Point		Nearest Locations
	User _j Stay Point		User Trajectory

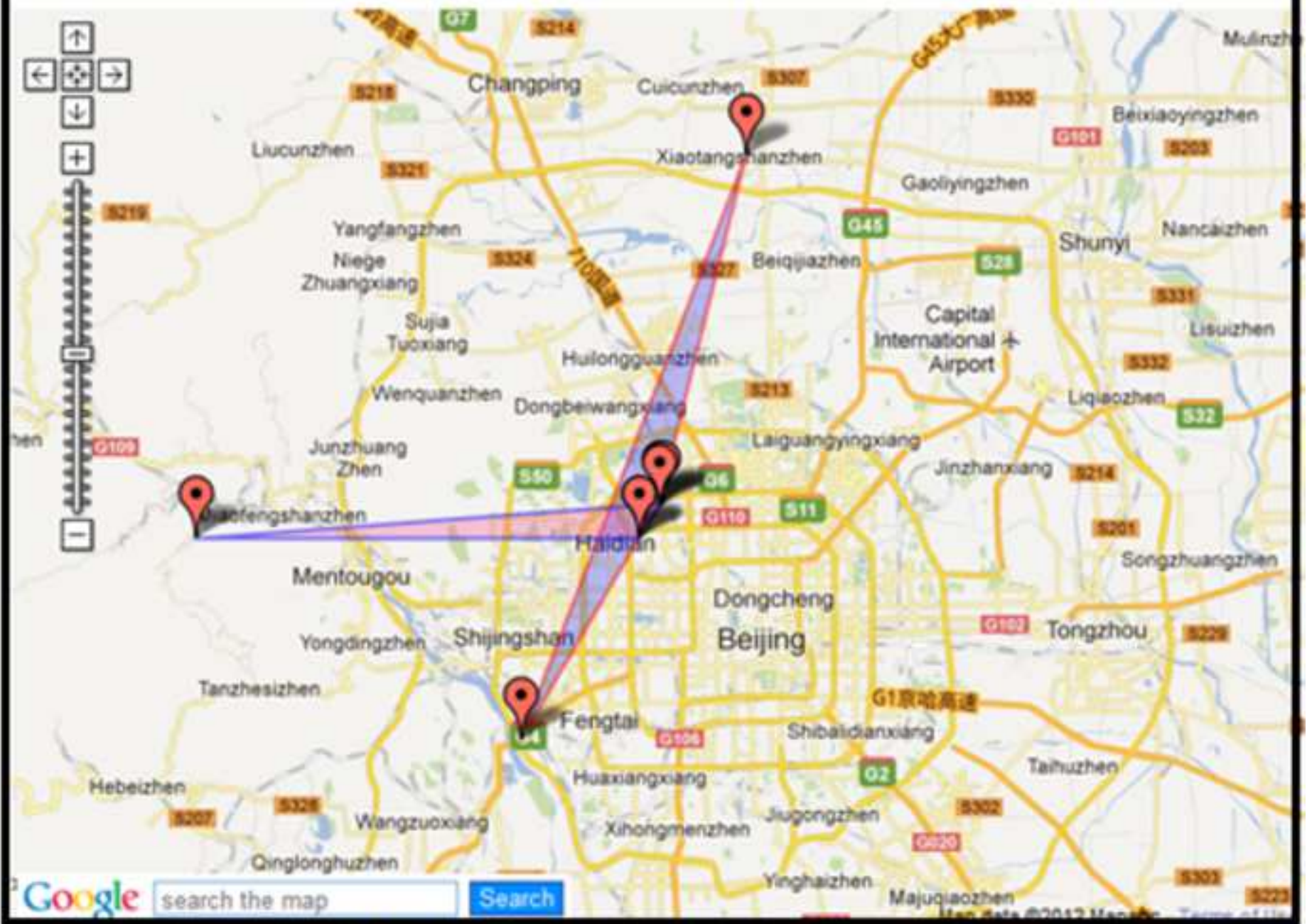




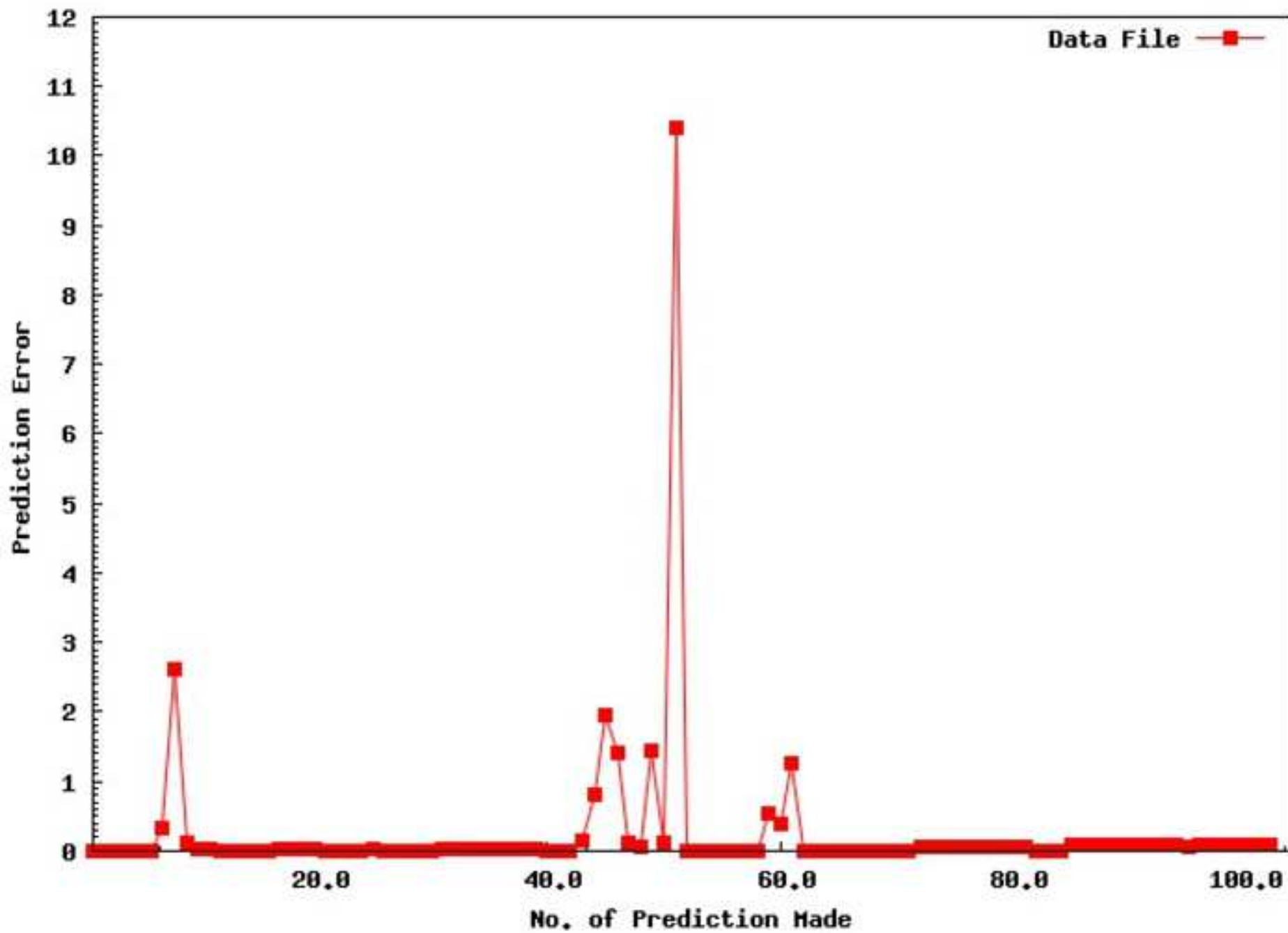


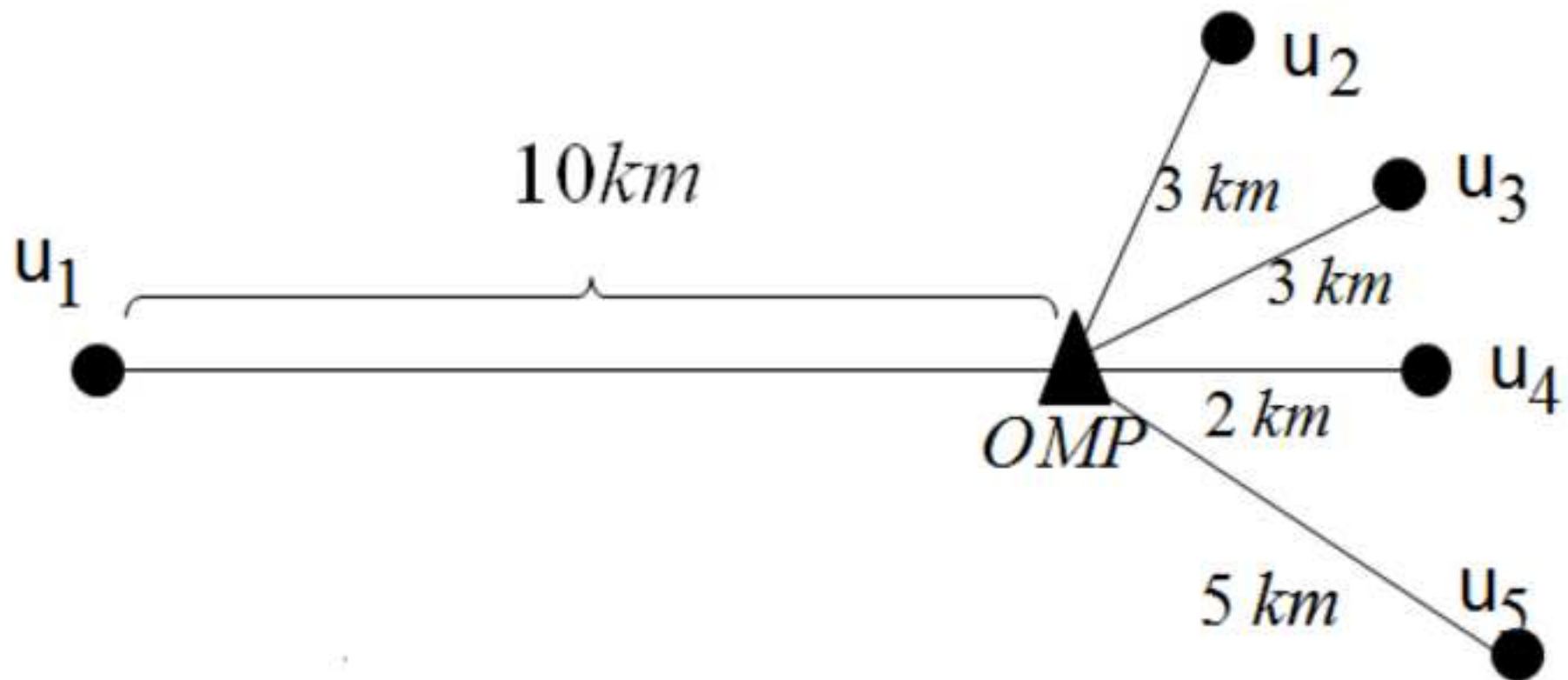


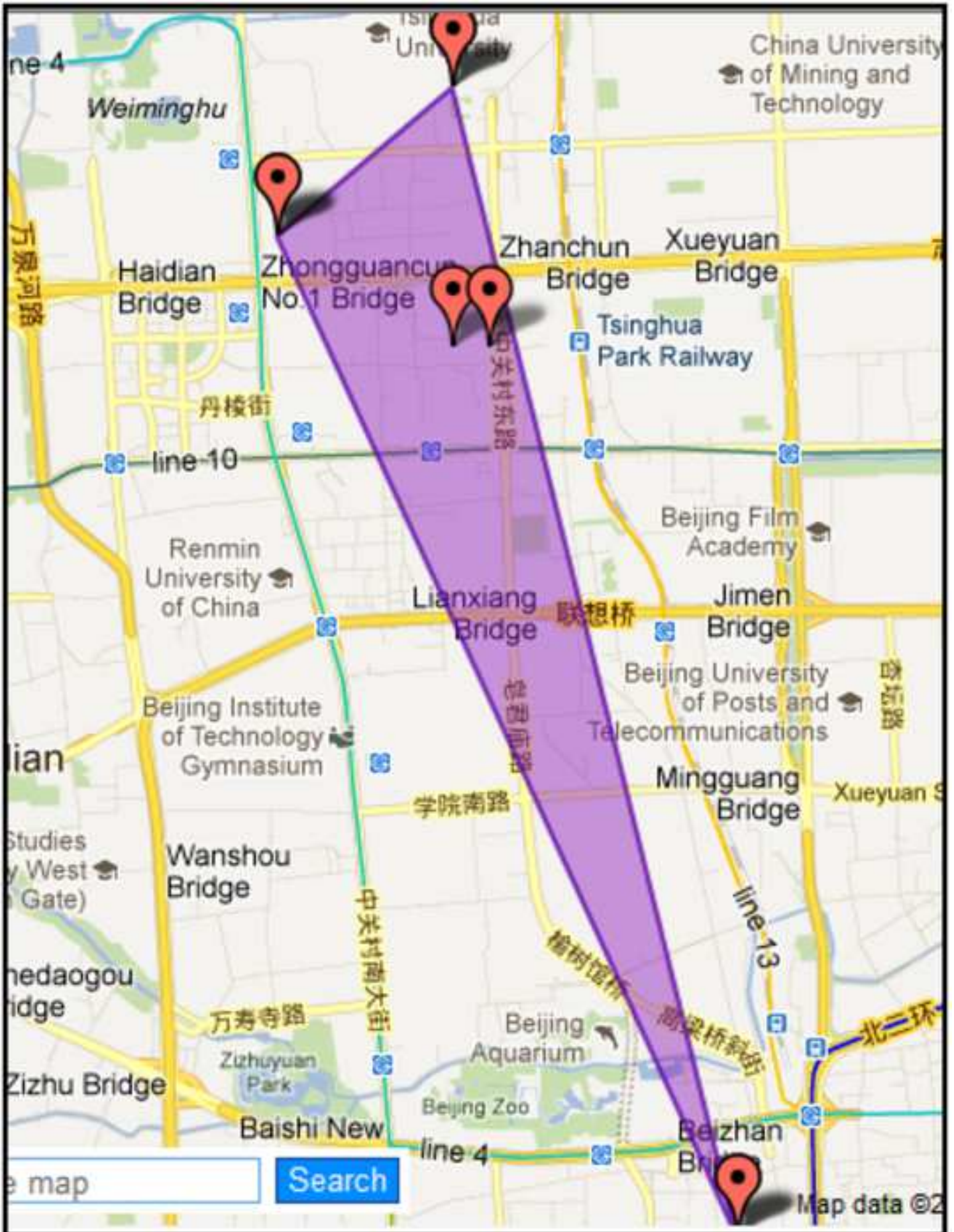
Convex Hull of Users Locations

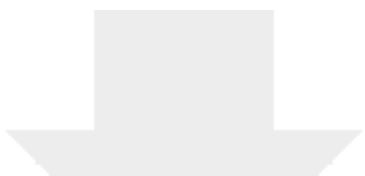


Location Point Prediction

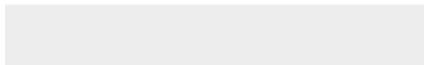







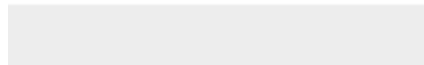


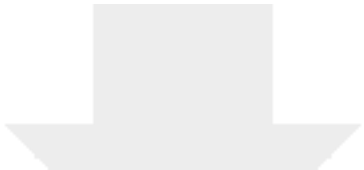
Click here to access/download
Supplementary Material
OMP_SNAM.tex





Click here to access/download
Supplementary Material
Responses to the reviewers Comments.pdf





Click here to access/download
Supplementary Material
OMP_SNAM.aux





Click here to access/download
Supplementary Material
OMP_SNAM.bbl





Click here to access/download
Supplementary Material
OMP_SNAM.bib





Click here to access/download
Supplementary Material
OMP_SNAM.blg

