A Trajectory-Based Recommender System for Tourism

Ranieri Baraglia[1], Claudio Frattari[2], Cristina Ioana Muntean[3], Franco Maria Nardini[1], and Fabrizio Silvestri[1]

[1] ISTI–CNR, Pisa, Italy
{name.surname}@isti.cnr.it
[2] University of Pisa, Pisa, Italy
claudiofrat2002@hotmail.it
[3] Babes-Bolyai University, Cluj-Napoca, Romania
cristina.muntean@econ.ubbcluj.ro

Abstract. Recommendation systems provide focused information to users on a set of objects belonging to a specific domain. The proposed recommender system provides personalized suggestions about touristic points of interest. The system generates recommendations, consisting of touristic places, according to the current position of a tourist and previously collected data describing tourist movements in a touristic location/city. The touristic sites correspond to a set of points of interest identified a priori. We propose several metrics to evaluate both the spatial coverage of the dataset and the quality of recommendations produced. We assess our system on two datasets: a real and a synthetic one. Results show that our solution is a viable one.

Keywords: tourist recommender systems, trajectory pattern mining.

1 Introduction

Recommendation systems deal with providing focused information to users that can likely be of their interest in a set of objects belonging to a specific domain (music, movies, books, etc.). Such systems are common in search engines and social networks, as well as in any situation where a focused suggestion might be of value. This work presents a recommender system that allows to provide personalized information about locations of potential interest to a tourist. The system generates suggestions consisting of touristic places, according to the current position of the tourist that is visiting a city and a history of previous visiting patterns from other users. For the selection of tourist sites, the system uses a set of points of interest (PoI) identified a priori. The system is structured into two modules: one operating offline and one operating online with respect to the current visit of a tourist. The offline one is used to create the knowledge model, that is then used for calculating suggestions. It is executed periodically when new GPS data are available for updating the knowledge model. The online one uses information from the current visit path of a tourist and the knowledge model to calculate a list of suggestions as possible next locations to visit.

R. Huang et al. (Eds.): AMT 2012, LNCS 7669, pp. 196–205, 2012.

2 Related Work

In [11], authors perform travel recommendations by mining multiple users' GPS traces. They model multiple users location histories with a tree-based hierarchical graph. Based on the graph, authors propose a HITS-based inference model, which regards the access of an individual in a location as a directed link from the user to that location. The recommendation process uses a collaborative filtering model that infers users' interests in an unvisited location based on her location histories and those of others. Results show that the HITS-based inference model outperforms baseline approaches like rank-by-count and rank-by-frequency.

Monreale *et al.* propose WhereNext, a method aimed at predicting with a certain accuracy the next location of a moving object [9]. The prediction uses previously extracted movement patterns named Trajectory Patterns, which are a concise representation of behaviors of moving objects as sequences of regions frequently visited with typical travel times. A decision tree, named T-pattern Tree [4], is built by using Trajectory Patterns and used as predictor of the next location of a new trajectory, finding on the tree the best matching path.

Kurashima *et al.* use the list of locations visited in the past by a user to incrementally build new trajectories that maximize their likelihood in the mixed topic-Markov model [6]. The best k routes satisfying a maximum time and distance constraints are returned.

Other works on geographical recommender systems use the concept of Point of Interest (PoI). In [3] a mobility-aware recommendation system, called PILGRIM, is proposed. It uses the location of a user to filter recommended links. The authors build models relating resources to their spatial usage pattern, used to calculate a preference metric when the current user is asking for resources of interest.

Lucchese *et al.* propose a novel random walk-based algorithm for the interactive generation of personalized recommendations of touristic places of interest based on the knowledge mined from photo albums and Wikipedia [8].

Our solution blends together user friendly approaches in the above mentioned literature, in order to produce exhaustive tourist recommendations. We expand trajectory mining from the simple use of GPS coordinates to the use of significant/relevant PoIs able to bring value to users according their interest. While WhereNext is able to produce significant predictions, we intent to deliver a broader solution: a recommendation system able to assist and offer not only the next step [9], but a list of suggestions from which the user can choose from.

3 The Proposed System

As Figure 1 shows, the architecture of our recommender system has two main modules: offline and online. The first one aims to create the knowledge model, which is the basis for computing suggestions. Its execution takes place when new data is available for updating the knowledge model. The online module uses the current user information and the knowledge model in order to produce a list of suggestions.

Building the Knowledge Model: The data processed by the offline module consists of a dataset of trajectories representing the movements of users in a certain period of time, as detected by their GPS devices, and a set of PoIs with their coordinates. The trajectories initially have the following format: $T =< (x_1, y_1, t_1), (x_2, y_2, t_2), \ldots, (x_n, y_n, t_n) >$, where (x, y) are the coordinates on the Cartesian plane, and t is the timestamp. To facilitate the trajectory mining process, we define the following distance function: $N : \Re^2 \rightarrow P(\Re^2)$, describing whether the points of a trajectory are related to a PoI. In order to do this, the Cartesian plane is divided into regions so that if a point falls in the region of a PoI, then this point is assigned to that region. This function allows us to simplify the process of mining, which manages trajectories expressed as: $T'' =< (A, t_1), (B., t_2), ..., (C, t_n) >$, where capital letters represent regions and t timestamps, instead of trajectories expressed as coordinates. It reduces the cardinality of the set of elements on which the knowledge model is computed. To divide the plane into regions [2] we exploited the QuadTree technique, which consists of a class of hierarchical data structures that have the common characteristic of recursive division of space [10]. The adopted QuadTree-based algorithm divides the plane into regions of the same shape, but of different sizes.

Regions can be very small where there are several close PoIs. Their size depends on the PoIs distribution, and especially on the distance from one PoI to another. The QuadTree technique is very efficient, it performs data comparison and data insertion with a complexity equal to $O(logn)$, where n are the number of PoIs in input. In the worse cases, corresponding to poor PoIs distributions, thus unbalanced trees, it can run in $O(n)$.

For constructing the knowledge model we first transform the dataset of trajectories of regions according to the QuadTree division of space. This allows us to identify points that fall in a specific region with identifier id. Thus the trajectories are represented as: $A \xrightarrow{\alpha} B$, where A and B are two regions and α is the estimated time needed to move from A to B. Then, to build the knowledge model the T-Pattern Tree is used. Frequent trajectories are trajectories with a support greater than a threshold value σ. The T-pattern Tree is built incrementally and

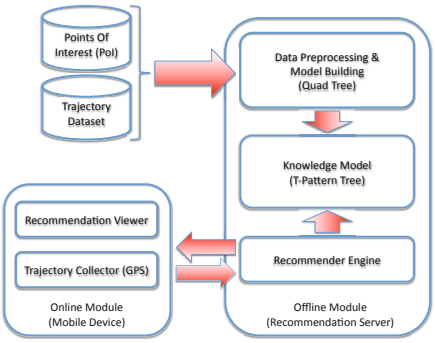

Fig. 1. Architecture of the proposed recommender system

trajectories having a prefix in common are overlapped on the tree to avoid unnecessary branches duplication. Each node is identified by a tuple $< id, region, support >$, where id is the node identifier, $region$ is the concerned region and $support$ is the sum of the supports of the various trajectories that have $region$ in that position. Each edge has associated a tuple $< [t_{min}, t_{max}], P(A \rightarrow B)] >$, where $[t_{min}, t_{max}]$ represents the estimated time needed to pass from a parent region to a child one, and $P(A \rightarrow B)$ indicates the probability of moving from region A to region B. The probability value is computed as the trajectories support value of a child node divided by the trajectories support value of its parent node. The T-pattern Tree can thus be modeled as a Markov chain and is used to find the similarity between trajectories frequently traveled and the current-analyzed one.

Computing Suggestions: User locations are obtained from GPS systems and sent to the offline module whenever a new position is detected. The recent movements of a user are used to set up the *current trajectory*, which is then compared with each practicable T-pattern Tree path. For each pattern match, a score, called *Punctual Score*, is calculated by assigning a value to each node of the current trajectory, which is then compared with those contained in T-pattern Tree. As in [9], the use of this score is designed to measure the reachability of a node r by a trajectory T which has already reached the parent node $r-1$. The punctual score indicates the matching of a node compared with the trajectory taken into consideration. The comparison may lead to three different cases for the calculation of the punctual score: 1) The current region is equal to the current node, and reached within the expected time. The punctual score is equal to the support associated to the node; 2) The current region is equal to the current node, but not reached within the maximum expected time. The punctual score is computed as: $^{node.support}/_{\beta * d_t}$, where β is a constant and d_t is the distance in time between the end interval and the time when the current region is reached; 3) The current region is not equal to that of the current node, the punctual score is computed as: $^{node.support}/_{(\beta * d_t + \alpha * d_s)}$, where α is a constant. Moreover, we specify the distance time tolerance th_t and distance space tolerance th_s as the maximum value that d_t and d_s can assume; when they exceed the specified values the punctual score of the current node is set equal to 0. The total pattern tree score $PathScore$ is computed as follows. Given a trajectory tr, a path $P = [p_1, p_2, ..., p_k]$ and a Punctual Score $PScore_k$ defined on each $p_k \in P$, the three indexes are computed as: 1) $AvgScore(tr, P) = \frac{\sum_1^n PScore_k}{n}$; 2) $SumScore(tr, P) = \sum_1^n PScore_k$; 3) $MaxScore(tr, P) = max\{PScore_1, \ldots, PScore_k\}$. AvgScore generalizes the concept of similarity by averaging the distance between the actual trajectory and the T-pattern Tree's pattern; SumScore is based on the concept of depth, the highest score is assigned to the longest path that intersects the actual trajectory; MaxScore considers only the node with the highest score based on the fact that if a trajectory has a good match with a node, it will probably have a good overall match. In our tests we use the SumScore method and assign the highest score to the longest path that intersects the trajectory. To carry out suggestions, candidates with higher PathScore and their children are returned,

indicating the next regions that one can get from the current position. Once the regions are found, we look for the associated PoIs, which are suggested to a tourist. For our solution, we create a knowledge model for the category of PoIs that represent tourist sites. Then, by default the user will receive suggestions on tourist places that are updated according to his moves. Moreover, by using the information associated to the edges we can also provide, as a function of the available means of transport, an approximate time and cost needed to reach a suggested place. There are three cases where the system fails to make a prediction: 1) The current trajectory is longer than each T-pattern Tree's trajectory; 2) The current trajectory is spatially too distant from any trajectory on the T-pattern Tree; 3) The current trajectory is temporally too far away from any trajectory on the T-pattern Tree. These events are directly dependent on the quality of the T-Pattern used. It is therefore necessary to evaluate a priori the predictive power of the set of T-Pattern used as the quality of the predictions depends on the spatial and temporal characteristics of this set. The T-pattern are sequences of spatial regions of different sizes and intervals of time. The sizes of these regions are the key to being able to produce the predictions of good quality. For example, considering only T-pattern covering a small portion of the total space, can not be processed predictions reliable.

4 Evaluation

We measure the effectiveness and the efficiency of the proposed solution by using two trajectory sets: *synthetic* and *real*, and a set of predefined PoIs. Moreover, the performance results obtained by the proposed solution were compared with those obtained by a greedy solution that carries out a list of suggestions made up of regions closer to the current location.

The synthetic dataset was created using a trajectory generator for a specific geographic area. It takes as input a dataset of PoIs, which are combined in sequences that form trajectories. The set of PoIs adopted during the tests include all the most important (monumental/artistic) PoIs in Florence, generated using information from Wikipedia. Through the Wikipedia API, we are able to retrieve the spatial coordinates of a list of PoIs in Florence.

By setting the parameters like the distance threshold d and the number of trajectories to build, it is possible to customize the dataset. This flexible mechanism allows us to generate a dataset of a predetermined size. Accordingly, for building a trajectory the following steps are required: i) we randomly select the starting point from 1022 PoIs extracted from Wikipedia, ii) we identify the starting point neighbors from the set of PoIs closer than d, iii) we rank the neighboring points by means of a function that minimizes the distance between the candidate PoI and the one currently analyzed; we select the next step in the trajectory according to the score associated to each candidate PoI - the higher the score, the higher the probability of selection, and iv) we terminate the trajectory building when the desired length is reached. The decision to model the interest for a PoI by the distance was influenced by the good results shown in [7]. The resulting synthetic dataset contains 20000 trajectories.

The real dataset is made up of data coming from Flickr. The trajectories are built using the photos submitted by users. A photo may have additional data such as the time it was taken and the geospatial coordinates of the object depicted. We considered the subject in the photos as potential real PoIs.

We built the dataset from information relating to photos taken in Florence from January 2004 to January 2010. With the data obtained it was possible to build daily trajectories for each user. The spatial coordinates associated with an individual photo may not exactly coincide with the PoI photographed. In order to build trajectories a data structure called R-Tree [5] was used which permits us to assign a rectangular area to each PoI belonging to the set of PoIs. If the coordinates of a PoI extracted from the dataset of photos fall into one of these rectangular areas, the representative corresponding PoI is assigned to the trajectory. Building the real dataset consists of three phases: 1) Build the R-Tree. Each node represents a geographical area and its children represent sub-areas. 2) Extract information from photos: user id, date and spatial coordinates. 3) Extract the PoI. For each pair of spatial coordinates contained in the list of PoIs from Flickr, we verify whether it is contained in one of the R-Tree's leaf nodes. If so, the PoI identifying that bounding box is added to the user's trajectory.

Evaluating the Quality of the Trajectory Set: The quality of the trajectory set is a key element for building the knowledge model with which the recommendations are computed. Therefore, it is important to understand in advance whether a set is valid for the effective evaluation of suggestions. To this end, in [9] the authors have proposed a method to establish a correspondence between the accuracy and the value of the support for a set of association rules. In our case, the ability to make accurate predictions also depends on the spatial characteristics of a set of T-pattern, not only on support. We refer to it as *Coverage*. The following indexes were adopted by us for assessing the Coverage:

- **SpatialCoverage** (SC) measures the fraction of the total space covered by the trajectory set as: $SpatialCoverage = \frac{\cup_{Tp \in Tp_{set}} Space(Tp)}{TotalSpace}$, where $Space(Tp)$ is the function that assigns to each T-pattern a portion of the plane that it fails to cover. $TotalSpace$ is the total space where tourists move around;
- **DataCoverage** (DC) defines the fraction of trajectories that go over the support value. This is computed as: $DataCoverage = \frac{|T| - Tpset}{|T|}$, where T is the trajectory set and $Tpset$ is the number of extracted trajectories which satisfy the support value.
- **RegionSeparation** (RS) measures the prediction accuracy as function of the prediction granularity. It is computed as: $RegionSeparation = \frac{MinimalRegion}{AVG_{r \in Tp \in Tp_{set}}}$, where $MinimalRegion$ is the minimum spatial granularity corresponding to a PoI within the considered space and $AVG_{r \in Tp \in Tp_{set}}$ is the average size of regions belonging to the trajectory set.
- **Rate** correlates all above three metrics as follows:
  $Rate = SpatialCoverage \cdot DataCoverage \cdot RegionSeparation$

As can be seen from Table 1, as the number of PoIs per region increases, the Rate value decreases. Even if the SpatialCoverage index increases, the other two indexes RegionSeparation and DataCoverage decrease. In fact, the higher values for Rate, i.e 0.06 for the real trajectory set and 0.08 for the synthetic trajectory set, are obtained when there are five PoIs per region. This is because RegionSeparation, and consequently Rate, rewards the correspondence between PoIs and regions. Almost identical Rate values were obtained for the same test with synthetical sets 5000 and 10000 trajectories. It shows that RegionSeparation is independent of the size of the trajectory set. The value of the support used to conduct the test is equal to 1.

Evaluating the Effectiveness and Efficiency: To evaluate the effectiveness of suggestions we adopted an empirical approach that estimates the percentage of errors in making recommendations using a test set [9]. The set of samples is divided into two disjoint subsets, a *training set* used to build the knowledge model (90%) and a *test set* (10%) used for evaluation. Each trajectory in the test set is iteratively divided into two parts: the first part represents the current trajectory on which we want to receive recommendations, and the second part is used for comparisons with the suggested regions. Initially, the current trajectory is represented by the first region of the analyzed trajectory and the remaining regions are used for comparison. A trajectory is divided in this way until the second part contains a single region. The tests to evaluate the efficacy of the proposed solution were conducted by computing a list of 10 regions as a suggestion. For evaluating the effectiveness, we adopted the following metrics:

- **Prediction Rate** (PR) is the percentage of trajectories for which the system is able to make a prediction.
- **Accuracy** (A) is the percentage of trajectories for which the system returns a list of suggestions containing the region that, in the test set, immediately follows the last region of the current trajectory.
- **Modified Accuracy** (MA) refines Accuracy. In [7] is shown that people tend to minimize the distance between locations. Accordingly, a tourist in a region may move to another PoI in the same region or in a different one. The region where the tourist is located is added to the suggestions.
- **Average Error** (AE) is the average error percentage computed for each trajectory. A trajectory of n regions is divided $n-1$ times, and $n-1$ comparisons are made. The result of each comparison is true, if the list of

Table 1. Coverage value for the real and synthetic trajectory sets

PoIs Region	Real Dataset				Synthetic Dataset			
	SC(%)	RS(%)	DC(%)	Rate	SC(%)	RS(%)	DC(%)	Rate
5	0,70	0,12	0,75	0,06	0,91	0,92	0,71	0,08
10	0,67	0,06	0,75	0,03	0,96	0,05	0,64	0,03
20	0,86	0,03	0,69	0,02	0,97	0,03	0,61	0,02
30	0,83	0,02	0,66	0,01	1	0,02	0,06	0,01

suggestions contains the next region, and false otherwise. Let a the number of false comparisons, the error rate for the related trajectory is equal to $a/n - 1$.

– **Omega** (Ω) measures the immediate utility of the generated suggestions [1]. It is computed as:

$$\Omega = \sum_{i=1}^{Ns} \frac{\sum_{k=1}^{n_k} [p_k \in \{S_i^{1,k} \cap R_i^{k+1,n_k}\}] \frac{f(k)}{n_k}}{N_s} \quad (1)$$

where N_s is the number of trajectories of the test set, n_k is the number of regions in the current trajectory, $f(k)$ is a function assigning a weight to a suggested region and p_k is a functions that returns 1 for a correct prediction and 0 on the contrary.

Table 2 shows the values of the metrics evaluated to measure the efficacy of the computed suggestions. They were computed by varying the number of PoIs per region and using a knowledge model built with the value of the support equal to 1. The number of PoIs in a region significantly affects the effectiveness of the system. Increasing the number of PoIs per region, the regions become larger and the prediction becomes easier because the knowledge model needs fewer examples to correctly predict the regions. Accordingly, the probability that a region is correctly suggested increases. The best value for MA and A is achieved with 30 PoIs per region, reaching a maximum value of 80.19% for MA, 66.21% for A, and 56.49% for PR. As can be seen from Table 2, AE decreases as the number of PoIs in the regions increases, leading to a higher accuracy in suggestions. The values for Ω indicate that the immediate utility of suggestions varies little by changing the number of PoIs per region. PR computed on the synthetic set is greater than the one on the real set.

Table 3 shows the performance evaluated by varying the support value σ, used to generate the knowledge model. Increasing σ, the percentage of trajectories correctly predicted decreases, revealing the non monotonic property of the support. Also, the number of trajectories used to build the knowledge model decreases and the model loses part of its predictive power. Even if for small values of the support PR is high enough, A never increases above 50%, instead decreases progressively. The support value that ensures the best A is equal to 1 for both the real and synthetic trajectory sets.

Concerning the real trajectory set, better results are obtained for $\sigma = 4$, when PR is equal to 36.18%. A for high support values is related to a small

Table 2. Effectiveness by varying the number of POIs per region

PoIs Region	Real Dataset					Synthetic Dataset				
	PR(%)	A(%)	MA(%)	AE(%)	Ω	PR(%)	A(%)	MA(%)	AE(%)	Ω
5	48,24	30,45	35,00	74,35	5,74	100	49,52	55,32	46,12	4,53
10	48,24	47,27	57,95	63,29	4,59	100	47,46	56,32	49,40	4,37
20	56,47	59,80	71,65	48,10	5,31	100	67,55	70,34	30,15	5,36
30	56,49	66,21	80,19	39,39	4,15	100	81,36	83,01	16,15	4,84

percentage of predicted trajectories, less than half of the available trajectories. In fact, a peak (67.27%) of the MA index correspond to only 36.18% of the total trajectories. Moreover, MA w.r.t A increases of about the 10%, it means that there are at least 10% of users move to a PoI within the same current region.

The AE values varies between 63.29% and 76.80%. It is important to note that this measure of error is only related to the percentage of suggested trajectories. Ω has a similar value for the first three tests, and then decreases to 1.31, showing that the suggestions related to the support value equal to 10 have the highest immediate utility. Ω is referred only to trajectories that have a correct prediction, so for the support value equal to 10, Ω refers only to 26.32% of the suggested trajectories.

Concerning the synthetic set, PR is always equal to 100%, A has a trend inversely proportional w.r.t. the support. The synthetic dataset maintains a PR greater than the one obtained on the real dataset.

The system efficiency was evaluated measuring the average elapsed time to compute a list of suggestions on a trajectory. This time depends highly on the cardinality of T-pattern Tree. The bigger and deeper the tree, the more the execution time grows and the cost of prediction rises. Tests were conducted by considering 100, 150, 200, and 300 requests per minute. In such tests the proposed system demonstrate to be able to respond quickly, with an average elapsed time of 1500 ms. For 100 requests per minute the average response time remains constant below 500 ms, then slightly increasing in the case of 150 requests per minute. Reasonable values are obtained also in the case of 200 and 300 requests per minute, with average response time values between 1500 and 2000 ms.

We present a comparison of the proposed solution versus a greedy solution, *Nearest*. Due to the scarcity of recommendation systems available and low availability of datasets used in other articles, we developed a simple recommender called Nearest, which returns a list of suggestions containing regions closest to the current region of the tourist, thus not adopting any process of mining.

Both methods, ours and Nearest, are evaluated on the synthetic dataset and best performance values for parameters, namely σ equal to 1 and number of PoIs per region equal to 30. In the case of our system we obtain a PR of 100%, an A of 81.36% and a MA of 83.01%, while Nearest obtained a PR of 100%, an A of 60.40% and a MA of 63.40%. Due to the fact that tests were conducted on

Table 3. Results varying the support and 10 POIs per region

σ	Real Dataset					Synthetic Dataset				
	PR(%)	A (%)	MA (%)	AE (%)	Ω	PR (%)	A (%)	MA (%)	AE (%)	Ω
1	48,24	47,27	57,95	63,29	4,59	100	47,46	53,32	49,40	4,37
2	48,24	43,18	53,41	64,68	4,57	100	42,70	48,15	52,63	3,91
4	36,18	51,51	67,27	63,82	4,18	100	37,76	42,30	57,42	3,50
6	36,18	45,45	61,82	71,32	2,71	100	33,74	38,41	61,10	3,11
8	36,18	36,36	53,94	76,80	1,93	100	30,92	35,21	63,75	2,73
10	36,18	26,36	46,06	63,82	1,31	100	28,80	33,16	67,65	2,27

synthetic data sets, PR for both system is 100%. However a clear performance of our system in respect with Nearest can be acknowledged from an increased A and MA.

5 Conclusions

We proposed a recommendation system that can assist a tourist visiting a city. It is able to generate suggestions of potential PoIs, depending on the current position of a tourist, and a set of trajectories describing the paths previously made by other tourists. The best effectiveness is achieved when the support σ is equal to 1 and when the number of PoIs per region is equal to 30. We also evaluated our proposed system against a simple baseline solution, which produces a suggestion list of regions closer to the tourist's current position. Results show that our solution clearly outperforms that one. We also proved that the response time enables it to be used interactively.

Our research has been funded by the POR-FESR 2007-2013 No 63748 (VISITO Tuscany) project and POSDRU/88/1.5/S/60185 (Investing in people!).

References

1. Baraglia, R., Silvestri, F.: An online recommender system for large web sites. In: Proc. of IEEE/WIC/ACM WI 2004. IEEE Computer Society, Washington, DC (2004)
2. Baraglia, R., Frattari, C., Muntean, C.I., Nardini, F.M., Silvestri, F.: Rectour: a recommender system for tourists. In: Proceedings of the First Workshop on Tourism Facilities (TF 2012) Colocated with the 2012 IEEE/WIC/ACM International Conference on Web Intelligence (2012)
3. Brunato, M., Battiti, R.: Pilgrim: A location broker and mobility-aware recommendation system. In: Proc. of PerCom, pp. 265–272. IEEE (2003)
4. Giannotti, F., Nanni, M., Pinelli, F., Pedreschi, D.: Trajectory pattern mining. In: Berkhin, P., Caruana, R., Wu, X. (eds.) KDD, pp. 330–339. ACM (2007)
5. Guttman, A.: R-trees: a dynamic index structure for spatial searching. SIGMOD Rec. 14(2), 47–57 (1984)
6. Kurashima, T., Iwata, T., Irie, G., Fujimura, K.: Travel route recommendation using geotags in photo sharing sites. In: Proc. CIKM, pp. 579–588. ACM (2010)
7. Lee, K., Hong, S., Kim, S.J., Rhee, I., Chong, S.: Slaw: A new mobility model for human walks. In: Proc. IEEE INFOCOM, pp. 855–863. IEEE (2009)
8. Lucchese, C., Perego, R., Silvestri, F., Vahabi, H., Venturini, R.: How Random Walks Can Help Tourism. In: Baeza-Yates, R., de Vries, A.P., Zaragoza, H., Cambazoglu, B.B., Murdock, V., Lempel, R., Silvestri, F. (eds.) ECIR 2012. LNCS, vol. 7224, pp. 195–206. Springer, Heidelberg (2012)
9. Monreale, A., Pinelli, F., Trasarti, R., Giannotti, F.: Wherenext: a location predictor on trajectory pattern mining. In: Proc. of KDD, pp. 637–646. ACM (2009)
10. Samet, H.: Hierarchical Spatial Data Structures. In: Buchmann, A., Smith, T.R., Wang, Y.-F., Günther, O. (eds.) SSD 1989. LNCS, vol. 409, pp. 191–212. Springer, Heidelberg (1990)
11. Zheng, Y., Xie, X.: Learning travel recommendations from user-generated gps traces. ACM TIST 2(1), 2 (2011)