Proceedings of **PerCom2003** - Fort Worth (TX), USA, March 2003

PILGRIM: A Location Broker and Mobility-Aware Recommendation System

Mauro Brunato Roberto Battiti

Dipartimento di Informatica e Telecomunicazioni
Università di Trento
via Sommarive 14 — I-38050 Povo (TN) — ITALY
E-mail: brunato|battiti@dit.unitn.it

Abstract

Mobile computing adds a mostly unexplored dimension to data mining: user's position is a relevant piece of information, and recommendation systems, selecting and ranking links of interest to the user, have the opportunity to take location into account.

In this paper, a mobility-aware recommendation system that considers the location of the user to filter recommended links is proposed. To avoid the potential problems and costs of insertion by hand, a new middleware layer, the location broker*, maintains a historic database of locations and corresponding links used in the past and develops models relating resources to their spatial usage pattern. These models are used to calculate a preference metric when the current user is asking for resources of interest.*

Mobility scenarios are described and analyzed in terms of possible user requirements. The features of the PILGRIM mobile recommendation system are outlined together with a preliminary experimental evaluation of different metrics.

1. Introduction

The ever growing share of mobile users in the networking scenario, made possible by the introduction of powerful palmtop devices and by the mushrooming of GPRS, 3G, Wi-Fi, Bluetooth and other wireless connectivity solutions, is pushing the need for new tools for mining the large amount of data provided by the Internet. Mobile computing, where users provided with PDAs, tablets or laptops are free to move while staying connected to the network, has proved to be a true revolution. The introduction of small pocket- and tablet-sized computers, and the contemporary blossoming of wireless networking solutions, from IEEE802.11b (aka Wi-Fi) to 3G cellular systems, is forcing a rapid change of paradigm in the area of service provisioning. In particular, the traditional notions that the user's location is fixed and relatively unimportant, and that the user can afford a lot of keytyping in order to access useful information, is not true anymore.

Many web pages are intrinsically related to physical locations, for example to shops, monuments, theaters and cinemas, restaurants, emergency facilities. Their interest for a mobile user is often dependent on his/her proximity. A tourist may ask "What next?", and expect a list of nearby places to see. A driver through the Death Valley may ask about the location of gas stations in order to minimize the risk of running out of fuel, or to minimize the expense (gas prices tend to grow towards the center). Unfortunately, today's Internet consists of an overwhelming amount of information, only partially organized by means of portals and search engines. Moreover, a typical search session requires multiple queries by the user, while portals need to be continuously updated, and tend to become dense and only readable on large screens.

The typical mobile user relies on small devices such as PDAs, with small-sized screens and slow on-screen keyboards, so neither bulky portal pages, nor keyword search engines are of much use. A recommendation system, taking into account location and other data, and providing a few links that are considered most interesting to the user in that particular context, would provide an agile and flexible environment for a mobile user. The context of pervasive computing in a wireless Internet framework is explored by the WILMA Project (Wireless Internet and Location Management) at the University of Trento (Italy). See the acknowledgments section for more detail.

In the following part of this paper, after a brief survey of currently available recommendation systems in Section 2 and of locality-based services in Section 3, the

PILGRIM (Personal Item Locator and General Recommendation Index Manager) location-aware recommendation system based on a *location broker* is presented and discussed in Section 4, where some preliminary simulation results are discussed. Finally, some problems and issues related to the usage of the system are discussed in Section 5.

2. Recommendation systems

A typical recommendation system [10] answers the question: "What are the k more interesting items for the current user?". To this purpose, user and item profiles are scanned and similarity techniques are employed for ranking all items in order to pick the most relevant ones.

User and item profiles need not necessarily be managed by site managers: techniques of *collaborative filtering* can be introduced where user profiles and evaluations are stored and used to automatically build a list of possibly attractive links specifically tailored for a particular user. Many recommendation systems, such as Tapestry [4] or Fab [2], require users to express their evaluation of the visited item, while others can gather implicit information. For example, the GroupLens [7] USENET news recommendation system uses reading times as a user interest measure.

Other system, such as PHOAKS [13], use data mining techniques to extract URLs or other information pointers from USENET postings or from bookmark collections.

A recommendation system is supposed to maintain a finite list of *users*, identified by unique IDs. Each user is associated to some *profile* information. A list of *items*, for instance web links, is also maintained along with relevant properties. The term *current user* will identify the user whom the recommendation list is being built for, and does not imply uniqueness: many "current users" may take advantage of concurrent instances of the recommender system at the same time.

Item ranking techniques are often based on comparison of user profiles (*user-based filtering*), which may include information provided by the user (his/her work, hobbies, last readings, and so on), or just a list of recently selected items. The current user profile is compared with all others, and the closest matches are used to build a plausible "top-k" item list. An example is the Amazon.com recommendation: "Users that bought this book also bought the following:". User profile comparison is a time-consuming procedure, and smart data structures need to be implemented in order to manage a large population.

A different class of recommendation systems is based on item comparison (*item-based filtering*) [12, 8, 6]: items are scanned, and each of them is evaluated via the question: "How relevant is this item for the user?". The question is answered through similarity with other items that were selected by the same user, and similarity between two items is in turn evaluated by considering how many users have selected both. Item profiles, taking into account explicit user evaluation, overall number of selections or the time of permanence of the user in the related web page, can also be considered in ranking. Many variants and combinations are possible between these two classes of algorithms.

3. Locality-based recommendation

Since the explosion of wireless networking, location has become a valuable piece of information in order to select relevant information to the user. Interesting work in this area is done in HP's *CoolTown* project (`http://cooltown.hp.com/`), with the purpose of mapping physical locations to Internet URIs. In this project, relevant locations are equipped with short-range infrared emitters that periodically broadcast their related URI to listening mobile devices that are pointed to them. The virtual extension of this project, *Websigns* [9], works by interfacing to a number of positioning systems without actually installing the beacons: the user's position, detected via GPS, is sent to a central server, which extracts all items whose direction and distance fall within some item-dependent intervals. The server sends the links to a client program on the user's PDA; a graphical front-end allows the user to choose a link and open a browser window.

A location-aware recommendation system should be able to produce a top-k items list for a given user whose location is known with a precision ranging from a few meters to some hundreds of meters. Position estimates can be obtained by means of many systems, such as GPS (outdoor only, with a precision of about 10m), active badges [14, 5] (precisions ranging from few centimeters to room size), or by exploiting the radio propagation properties of the wireless networking medium [1, 11, 3] (with precisions of few meters in the Wi-Fi case). The latter solution is of particular interest because it does not need additional infrastructure, and the normal networking equipment is used both for communication and for location detection.

4. The PILGRIM system

A mobile user is likely to handle the PDA only for the time that is strictly needed to find an interesting link and follow it: being possibly in a public place, she is not willing to fumble with the device in order to give an explicit evaluation of the chosen item. So, only implicit information about the choice can be gathered:

- Was the presented item clicked or not?
- How long has the page remained on screen?

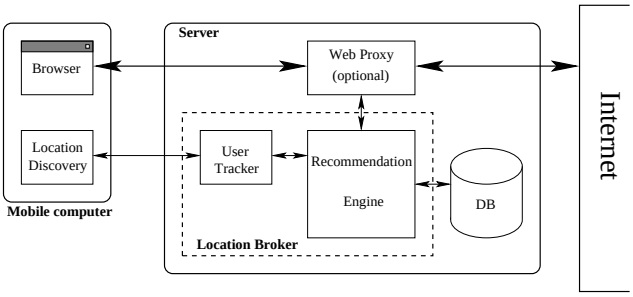

Figure 1. Architecture of the PILGRIM system.

- What was the subsequent action of the user (she abandoned the site, or she visited also linked pages)?

In a mobile environment, however, another important piece of information is the following:

- What was the user *position* when she clicked the link?

The purpose of the PILGRIM system is to integrate information about the current user location (possibly also the previously followed track) into traditional recommendation systems.

4.1. Architecture of the system

The PILGRIM system is structured as an automated learning component to develop models relating resources to their spatial usage pattern by mining the historic database that records past accesses to sites. The models are finally used to generate a recommendation list.

The basic building blocks of the system are shown in Figure 1. On the client side, possibly a PDA with low computing speed, two components are active. The first is the normal off-the-shelf Internet browser, and it is the only component that the user sees on the screen during normal operation. The second component, the *location discovery* application, is a small process that enables the PDA to obtain positioning data and to send them to the server; for instance, radio signal strength from surrounding Wi-Fi access points or raw GPS data can be measured. This module is mostly transparent to the user. It will only display a startup dialog for initialization purposes, for example to change privacy settings (see Section 5.3). The two components are independent: the system could take advantage from an integrated solution, but this may not be applicable to all systems. For instance, many lightweight browsers in use on PDAs do not allow component technologies such as Java or ActiveX, and even scripting languages may not be supported.

The location discovery application running on the client sends position updates to the server-side *location broker*. This is in turn composed of two components. The first, the *user tracker*, is in charge of computing the location data transmitted by the client in order to obtain a good estimate of the user position (due to power and CPU limitations, it may be impractical for the PDA to compute the precise location, and only raw data are transmitted to the server) and to track the user's movement. The second component, the *recommendation engine*, is the core of the system: it maintains the access database, containing data about what links have been followed by the user, and from what physical position. These data, together with the user's location provided by the user tracker module, are employed to generate a list of possibly interesting links.

4.2. Collaborative filtering and ranking procedure

Once the database is populated with past user accesses to items, its data can be used to build a model of user preference. Thus, the chosen approach considerably differs from other systems such as Websigns, where the database is updated and maintained by hand, and is more similar to the collaborative filtering paradigm, where the quality of recommendations shapes up as long as users interact with the system.

The model of the user will be expressed in terms of a metric, so that location-dependent item ranking is based on the past user choices. In this section, a metric based on inertial ellipsoids is introduced.

The recommendation engine works on a set of s links, each identified by a unique id $l = 1, \ldots, s$. Suppose that site l has been visited N_l times (possibly by different users), and let the set of points $P_i^l = (x_i^l, y_i^l)$, $1 \leq i \leq N_l$, represent the N_l physical locations where link l was clicked. A locality measure of link l can be obtained by calculating the *inertial ellipsoid* of its points. Points can be associated to a "mass" that is related to the level of trust of the received feedback. In the current version, for simplicity, all points are physically modeled as unit masses. The inertial ellipsoid has the following quadratic equation:

$$\begin{pmatrix} x - \bar{x_l} & y - \bar{y_l} \end{pmatrix} M_l^{-1} \begin{pmatrix} x - \bar{x_l} \\ y - \bar{y_l} \end{pmatrix} = 1,$$

where $\bar{x_l}$ and $\bar{y_l}$ are the coordinates of the center of mass, while matrix M_l is the second-order moment matrix (the covariance matrix):

$$\bar{x_l} = \frac{1}{N_l} \sum_{i=1}^{N_l} x_i^l, \qquad \bar{y_l} = \frac{1}{N_l} \sum_{i=1}^{N_l} y_i^l,$$

$$M_l = \frac{1}{N_l} \begin{pmatrix} \sum_{i=1}^{N_l} (x_i^l - \bar{x_l})^2 & \sum_{i=1}^{N_l} (x_i^l - \bar{x_l})(y_i^l - \bar{y_l}) \\ \sum_{i=1}^{N_l} (x_i^l - \bar{x_l})(y_i^l - \bar{y_l}) & \sum_{i=1}^{N_l} (y_i^l - \bar{y_l})^2 \end{pmatrix}.$$

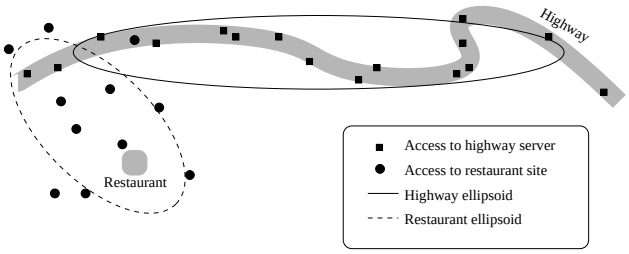

Figure 2. Sites with different access metrics.

Being positive definite, the matrix M_l^{-1} defines a distance between points $P = (x_P, y_P)$ and $Q = (x_Q, y_Q)$:

$$d_l(P,Q) = \begin{pmatrix} x_P - x_Q & y_P - y_Q \end{pmatrix} M_l^{-1} \begin{pmatrix} x_P - x_Q \\ y_P - y_Q \end{pmatrix}. \quad (1)$$

Note that the distance function d_l is indexed with the site number. This highlights the fact that in our model every site has a different metric.

Let $\bar{P}_l = (\bar{x}_l, \bar{y}_l)$ be the center of mass for site l. The distance d_l can be used as a measure of interest of site l for a user located at position $P = (x, y)$. Let us define the *preference* for site l at point P as

$$r_l(P) = \frac{1}{d_l(P, \bar{P}_l)},$$

so that site l is preferable to site l' at point P if $r_l(P) > r_{l'}(P)$ (preference is $r_l(P) = +\infty$ on the center of mass).

The set of preference functions $(r_l)_{1 \le l \le s}$ induces at every point P a permutation $\pi^P = (\pi_1^P, \ldots, \pi_s^P)$ of the site IDs having the property

$$\forall i \in \{1, \ldots, s\} \qquad r_{\pi_i^P}(P) \ge r_{\pi_{i+1}^P}(P).$$

The permutation is uniquely defined modulo equalities of the preference function; in this case, any tie-breaking rule, such as ID order, properly defines a unique permutation:

$$\forall i \in \{1, \ldots, s\} \qquad r_{\pi_i^P}(P) = r_{\pi_{i+1}^P}(P) \Rightarrow \pi_i^P < \pi_{i+1}^P.$$

The advantages of the ellipsoid metric with respect to simpler techniques can be understood by referring to Figure 2. Consider two candidate links. The first corresponds to a highway information server: it contains up-to-date information about the status of a highway segment, and is mostly used by people driving along that road. User access locations are represented by solid black squares. Almost all accesses to the site have been performed along the highway. Because of the unidimensionality of the road, there is a strong correlation between the x and y coordinates of the points, and the resulting ellipse, that with the solid outline, has high eccentricity. Its preference function, $r_{\text{highway}}(P)$ decreases slowly when moving from the average access position along the highway, while it drops very rapidly when moving outside the road. On the other hand, a restaurant placed nearby the highway, but not directly accessible, has a less eccentric region of interest (the small black circles). The resulting ellipse, with a dashed outline, is less eccentric, even though it still shows a preferential direction, due to the physical visibility of the building, or to the terrain morphology. The preference function, $r_{\text{restaurant}}(P)$, decays more regularly with distance from the center.

Note that the center of the ellipse does not coincide with the restaurant. In fact, no information is built in the system, and the geographical relevance of a link is gradually inferred through the ellipsoid metric just described: every time a user clicks a link, the recommendation engine updates the database, while inertial ellipsoids are periodically updated on the basis of the database records. The fact is even more apparent in the highway case, where the actual server could be located elsewhere in the country.

For the above reasons, the ellipsoid metric is a promising technique for a location-aware recommendation system, and it was chosen for the `PILGRIM` system. The following alternative candidate metrics have been considered.

- **Euclidean distance** (from the center of mass): in this case, only the Euclidean distance

  $$d^2(P,Q) = (x_P - x_Q)^2 + (y_P - y_Q)^2$$

  is used, so that sites are ranked according to the distance of their centers from the current position. Unlike the ellipsoid distance (1), the metric is the same for each site, so no index l is used.

- **Isotropic distance** with radial multiplier: distance is divided by the second-order momentum of distances from the center of mass:

  $$d_l^2(P,Q) = \frac{d^2(P,Q)}{\frac{1}{N_l}\sum_{i=1}^{N_l} d^2(P_i^l, \bar{P}_l)};$$

  this corresponds to the ellipsoid metric if accesses to sites are isotropic.

- **Nearest-neighbor ranking**: the link having been accessed from the nearest recorded point to P is ranked at the first place for point P, and so on. This technique is not very scalable, as it slows down while the number of accesses increases. Moreover, sporadic accesses far from the preference region tend to persist, instead of being outweighed by the remaining points.

Figure 3 shows the results of comparisons of the above described metrics in a simulated environment. Ten sites

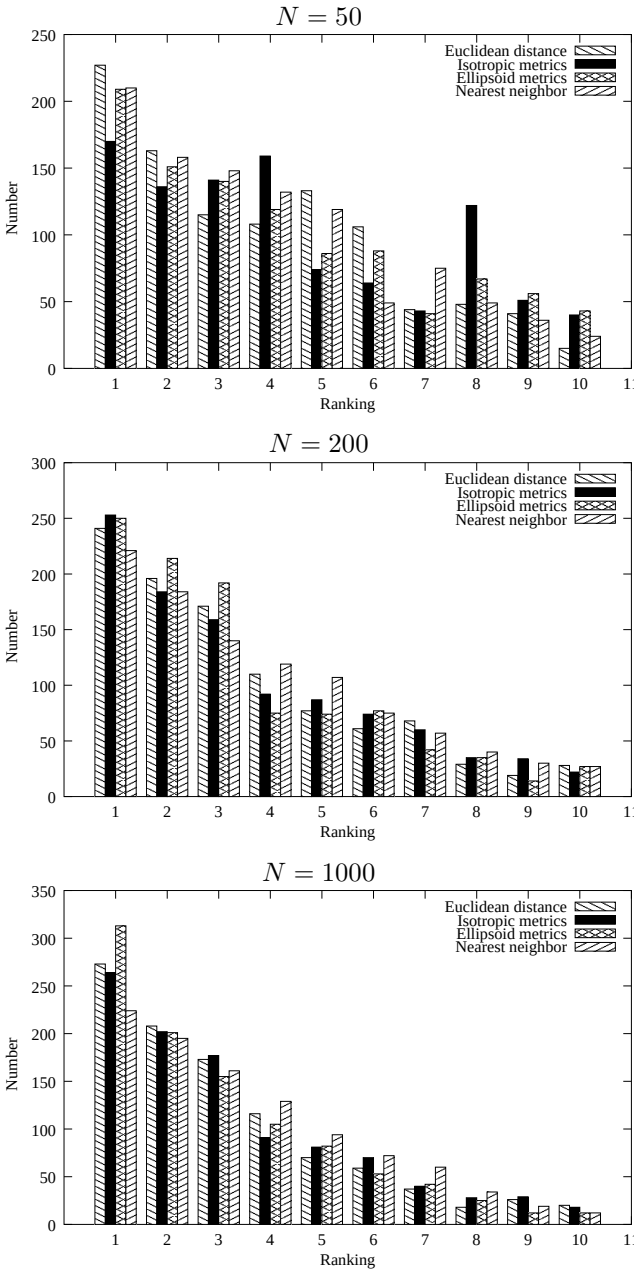

Figure 3. Distribution of ranking orders.

Table 1. Average ranking order of the preferred item.

Algorithm	$N = 50$	$N = 200$	$N = 500$
Euclidean distance	**2.842**	2.496	2.213
Isotropic metrics	3.391	2.571	2.333
Ellipsoid metrics	3.122	**2.356**	**2.05**
Nearest neighbor	2.868	2.729	2.498

have been placed in random positions across a 1000m $\times$ 1000m square. To every site, a random ellipse is associated and used to generate access instances (i.e., user clicks). This ellipse is unknown to the location broker.

The ellipses have been used to generate a database of N access instances (where $N = 50, 200, 1000$). After the database has been generated, 1000 tests have been performed where a random site l is chosen and a user is placed in a random position P with Gaussian distribution around the site location (the center of its random ellipsoid). Next, it is assumed that a user at location P is interested in site l and wishes to find it in the topmost ranking position of the recommendation list. The user position P is used to generated four different recommendation lists, one for every candidate metric, and the ranking order of site l in each of these lists is recorded.

The histograms report the ranking distribution of the sites chosen by simulated users. For example, the bottom histogram shows that, in more than 300 cases out of 1000, the site preferred by the user has been put at the top of the list based on Ellipsoid distance. Note that the Ellipsoid metric distribution tends to concentrate towards the left for a large historic database. The ideal situation (where all chosen sites are ranked first) cannot be obtained. This is a feature of the model intended to represent the intrinsic level of uncertainty of the user choice.

Results are synthesized in Table 1, where the average ranking of the preferred site is shown for all metrics and database sizes. For a small number of database entries, a simple Euclidean distance model seems more appropriate, while the ellipsoid model, being characterized by more parameters, achieves a better representation, but requires more samples.

4.3. Implementation

At its first implementation stage, the system is being currently tested as a one-page web application. A screenshot of the system is shown in Figure 4. The different building blocks shown in Figure 1 are implemented as separate C++ classes (the location broker and the recommendation

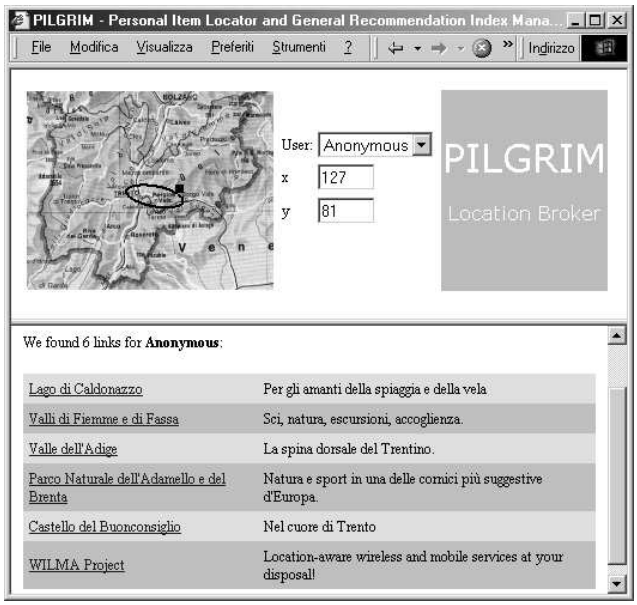

Figure 4. The experimental `PILGRIM` system.

index generator) and collected into one ActiveX component working also as position display (the top left map in Figure 4). The component, written in C++ with the Microsoft Foundation Classes library, interacts with the standard HTML form in the top frame of the browser, also showing the user name and coordinates, to generate the recommendation page in the bottom frame. If the mouse pointer passes over a link in the bottom frame, the corresponding ellipse is shown in the map.

In order to have the system work with a small number of actual users, to avoid small-number statistical fluctuations, the actual inertial ellipsoid is compensated by averaging with a fixed-radius circle having the same center. Let r be a default radius, for instance 1 kilometer; then

$$N_r = \begin{pmatrix} r^{-2} & 0 \\ 0 & r^{-2} \end{pmatrix}$$

is the matrix of the quadratic form associated to the circle of radius r. The actual matrix used for the evaluation of the ellipsoid metric of link l is the weighted average

$$M_l' = w_{N_l} N_r + (1 - w_{N_l}) M_l^{-1},$$

where the weight of the circle $w_{N_l} \in [0, 1]$ tends to 0 as the number of accesses to link l increases. In the current implementation,

$$w_n = e^{-\frac{n^2}{k}},$$

with k depending on the problem scale and on the desired convergence rate.

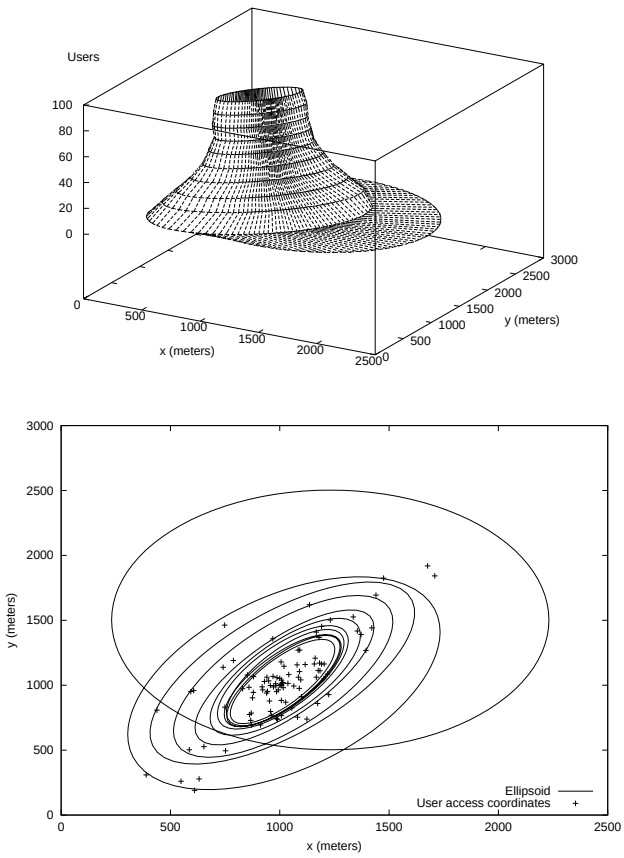

Figure 5. Compensated ellipsoid evolution.

Figure 5 shows the evolution in time of the compensated inertial ellipsoid representing the spatial distribution of user accesses to a site. User accesses are added one at a time to the database. The three-dimensional graph shows the evolution of the ellipsoid when the circle radius is $r = 1000$m and the weight parameter is $k = 500$: at the beginning, when the site is not yet very popular, the circular default prevails; later, the actual inertial ellipsoid outweighs the default circle and the correct shape is reached. Note that the initial estimate, the fixed-radius circle, is only based on the first access, and it can be off-center with respect to the overall distribution. The two-dimensional graph provides a view from the top, and also shows the actual position of the user locations.

5. Issues and problems

5.1. Scalability

A problem with all database applications is scalability: a service working well on a small test scale may

not work when deployed in a larger context. Basic ranking algorithms require scanning a large portion of the item database in order to build a list, and various nested scans could be necessary to sort it.

Smart locality-based data structures, such as quad-trees, R-trees and others, can ease the work; however, the system should be able to exploit its own locality-based nature and it should be implemented as a distributed system where local databases contain information about items with a strong localization, and a peer-to-peer content distribution scheme enables synchronization among all local servers.

5.2. Malicious ranking modifications

The proposed ranking procedure is highly democratic: any user implicitly "votes" and evaluates a link by taking appropriate action. However, a site administrator can easily use this system at her own advantage by repeatedly selecting the link to the site, by extensively browsing it from different locations in order to enlarge the corresponding inertial ellipsoid as much as possible, and so on.

Therefore, user preference should be managed very cautiously. Possible courses of action are:

- Store only the last visit from each user. This strategy is not recommended, because an interesting site is likely to be browsed extensively on the *first* visit, which could be regarded as more meaningful that the subsequent ones. In fact, once the user knows the site structure, she will directly get to the point without spending time in surfing.

- Store only the first visit. In this case, only the user's "first impression" is recorded. While this is meaningful information, subsequent accesses are important to establish the real value of the site.

- Some sort of average visit information is maintained and updated.

However, the number of visits from the same user is a very important evaluation issue, therefore it is necessary to evaluate whether the bias induced by malicious repeated accesses is so great to justify drastic countermeasures.

5.3. User privacy

The introduction of location- and profile-based systems encounters significant resistance from the public because of privacy concerns. In particular, tracking a user's position with precision of a few meters may be considered too harmful to implement. While perfect privacy cannot be ensured for any system, the architecture shown in Figure 1 can implement some privacy-enforcing schemes.

If the user is worried about communicating her own location, the location discovery procedure on the mobile client can be set in order to add noise to the data that shall be transmitted to the location broker. In this case, the server's response may be less accurate. However, the location procedure can be executed locally by the client, while a noisy version is sent to the server. In this case, the server may respond with a wider range of choices, to be refined by the client by using the exact position. This option requires a larger amount of communication and more CPU utilization by the client, so that a tradeoff among user privacy, response accuracy and battery consumption must be sought.

The system requires the user to be identified by some ID, in order to match the information coming from the location discovery module with the queries coming from the browser. If the user does not want to be identified, a one-time session ID can be used in place of the user ID.

Clearly, these options require the user to trust the system: the average user cannot check whether noise is actually added, or that the actual ID is never transmitted, or that the server is not trying to match the MAC address of the wireless network card with the session ID to build a persistent user history. The source code is going to be released for research purposes as soon as it exits its preliminary phase, in order to allow communities to check the privacy requirements.

Another field where the `PILGRIM` system may prove useful is that of *emergency management*, where the coordination of rescuing teams can benefit from location information and from preference rankings inferred during periodic disaster simulations. In this context, like in other mission-oriented applications, privacy issues are not relevant, and accurate identification is a desirable feature.

6. Conclusions

In this paper the novel concept of *location broker* as an independent entity mediating between mobile users and sites of interest has been presented and its introduction has been motivated. A system for location-aware recommendation of web links has been outlined. The architecture of the system and the core ranking algorithm have been described.

Current work is focusing on integration with currently available recommendation systems and on extensive real-world tests in different contexts.

Acknowledgments

This research is partially supported by the Province of Trento (Italy) in the framework of the WILMA (Wireless Internet and Location Management) project (`http://www.wilmaproject.org/`).

References

[1] P. Bahl and V. N. Padmanabhan. RADAR: An in-building RF-based user location and tracking system. In *Proceedings of IEEE INFOCOM 2000*, pages 775–784, Mar. 2000.

[2] M. Balabanovič and Y. Shoham. Fab: Content-based, collaborative recommendation. *Communications of the ACM*, 40(3):66–72, Mar. 1997.

[3] R. Battiti, M. Brunato, and A. Villani. Statistical learning theory for location fingerprinting in wireless LANs. Technical Report DIT-02-0086, Dipartimento di Informatica e Telecomunicazioni, Università di Trento, Oct. 2002.

[4] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, Dec. 1992.

[5] A. Harter, A. Hopper, P. Steggles, A. Ward, and P. Webster. The anatomy of a context-aware application. In *Proceedings of MOBICOM 1999*, pages 59–68, Aug. 1999.

[6] G. Karypis. Evaluation of item-based top-n recommendation algorithms. Technical report, University of Minnesota, Department of Computer Science / Army HPC Research Center, Minneapolis, USA, 2000.

[7] J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon, and J. Riedl. GroupLens: Applying collaborative filtering to USENET news. *Communications of the ACM*, 40(3):77–87, Mar. 1997.

[8] M. O'Connor and J. Herlocker. Clustering items for collaborative filtering. Technical report, University of Minnesota, department of Computer Science, Minneapolis, USA, 2000.

[9] S. Pradhan, C. Brignone, J.-H. Cui, A. McReynolds, and M. T. Smith. Websigns: Hyperlinking physical locations to the web. *IEEE Computer*, pages 42–48, Aug. 2001.

[10] P. Resnik and H. R. Varian. Recommender systems. *Communications of the ACM, special issue*, 40(3):56–58, Mar. 1997.

[11] T. Roos, P. Myllymäki, H. Tirri, P. Misikangas, and J. Sievänen. A probabilistic approach to WLAN user location estimation. *International Journal of Wireless Information Networks*, 9(3), July 2002.

[12] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of WWW2001*, May 2001.

[13] L. Terveen, W. Hill, B. Amento, D. McDonald, and J. Creter. PHOAKS: a system for sharing recommendations. *Communications of the ACM*, 40(3):59–62, Mar. 1997.

[14] R. Want, A. Hopper, V. Falcao, and J. Gibbons. The active badge location system. *ACM Transaction on Information Systems*, 10(1):91–102, Jan. 1992.