Technical report DIT-02-0093, Università di Trento, November 2002

A Location-Dependent Recommender System for the Web

Mauro Brunato Roberto Battiti Alessandro Villani Andrea Delai
DIT — Dipartimento di Informatica e Telecomunicazioni
Università di Trento — via Sommarive 14, I-38050 Povo (TN) — ITALY
brunato|battiti|avillani|delai@dit.unitn.it

***Abstract*— A relevant piece of information in many context-aware applications for wireless and mobile users is the user's current location. Knowledge of the position, when combined with the user preferences, permits efficient service (or product) location, location-dependent alerting, and location-aware recommendation systems.**

We propose a recommendation system that is based on a standard web browser and where models determining the relevance of a given URL in a given region are derived in an automated and adaptive way through the collaboration of users of the system. With respect to existing location-dependent recommendation systems, the advantage of our proposal lies in the reduced effort required for system development and in the increased independence of the recommendation from the services (or products) owners. After an initial tuning phase, a specific URL will be recommended to a user in a given location in a way that considers where and how often it was accessed by the previous users.

In detail, a new middleware layer, the *location broker*, collects a historic database where user positions and links used in the past are analyzed to develop models relating resources to their spatial usage pattern and to calculate a preference metric when the current user is asking for recommendations.

The focus of this work is on scalability issues. When the system is used in a wide area (possibly covering a sizable fraction of the entire web), the size of the database and the complexity of the models increase very rapidly. In an ubiquitous computing scenario where a multitude of wirelessly interconnected system surround a mobile user, the number of resources in an explored region may easily grow to contain thousands or millions of items. We describe a suitable data structure that permits scalability and analyze the empirical computational complexity both on a simulated scenario and in a real-world context in our province[1].

***Index Terms*— Mobile devices, Context-aware computing, Recommender systems, Collaborative filtering, Spatial databases**

I. Introduction

There is a growing consensus that the new tools for supporting mobile and wireless applications have to be developed in a user-centric way. Unless the cognitive burden required to receive the information that is appropriate in a given context is substantially reduced, the average user is not willing to spend time in tuning the system, selecting information from a small output device and entering large amounts of data through cumbersome interfaces. Adaptive context-aware system need to develop models of the relevance of different resources (e.g. URLs) for a specific user, and to filter the information so that only a small selection of relevant and limited resources is presented.

The location is of course a critical component of the context for mobile users. The introduction of small pocket- and tablet-sized computers, and the contemporary blossoming of wireless networking solutions, from IEEE802.11b (a.k.a. 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 key-typing in order to access useful information, is not true anymore. A recommendation system, taking into account location and other data, and providing a few links that are considered most interesting to the user in a particular context, would provide an agile and flexible environment for a mobile user. This system can be an ideal complement to methods providing dynamic customization of content for wireless clients [1], once the relevant links are selected.

Location-aware mobile commerce systems are considered for example in [2], location-aware shopping assistance is described in [3]. A mapping of physical locations to Internet URIs is proposed in HP's *CoolTown* project[2]. Relevant locations are equipped with short-range infrared emitters that periodically broadcast their related URI to listening mobile devices. The virtual extension of this project, *Websigns* [4], 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. The context of pervasive computing in a wireless Internet framework is also explored by our WILMA Project[3] (Wireless Internet and

[1]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/).

[2]http://cooltown.hp.com/
[3]http://www.wilmaproject.org/

Location Management) at the University of Trento, where the PILGRIM location-broker and mobility-aware recommendation system has been recently proposed. This papers deals with scalability issues to make the system available for a global use in the Internet, where the number of resources in the system database may easily grow to reach millions of items.

The rest of the paper is organized as follows. In Section II, a short survey of currently available recommender systems is presented. In Section III a model for describing the preferences of mobile users is presented, together with an overview of the architecture of the PILGRIM (Personal Item Locator and General Recommendation Index Manager) system. In Section IV the ER-tree, a spatial indexing structure tailored for the requirements of the PILGRIM system, is introduced and motivated. Section V is devoted to experimental evaluation of the ER-tree structure in simulated random environments. Finally, in Section VI conclusions are drawn.

II. Recommendation systems

A typical recommendation system [5] answers the question: "What are the k more interesting items for the current user?" For this purpose, user and item profiles are scanned and similarity techniques are employed to determine the most relevant items.

Techniques of *collaborative filtering* can be introduced where user profiles and evaluations are stored and used to automatically build a list of links specifically tailored for a particular user. Many recommendation systems, such as Tapestry [6] or Fab [7], require users to express their evaluation of the visited item, while others can gather implicit information. For example, the GroupLens [8] USENET news recommendation system uses reading times as a user interest measure. PHOAKS [9] uses data mining techniques to extract URLs or other information pointers from USENET postings or from bookmark collections.

A recommendation system maintains 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.

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. 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*) [10], [11], [12]: 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.

III. A location-aware model of user preferences for web sites

The PILGRIM recommendation system [13] generates recommendation lists based on the user location and on web site information gathered from previous usages of the same sites by other people. For completeness we briefly summarize the PILGRIM approach in the current section.

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], [15] (precisions ranging from few centimeters to room size), or by exploiting the radio propagation properties of the wireless networking medium [16], [17], [18] (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.

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 and may not be 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?
- What was the subsequent action of the user (she abandoned the site, or she visited also linked pages)?

In a mobile environment, however, another crucial 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 into traditional recommendation systems in an adaptive way.

A. 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 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. This

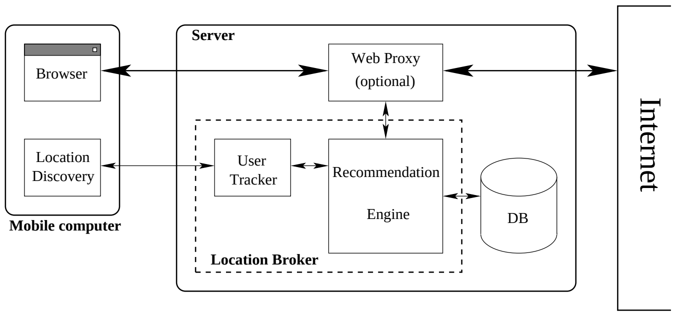


Fig. 1. Architecture of the PILGRIM system.

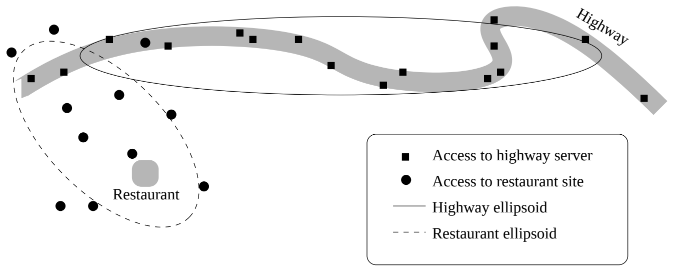


Fig. 2. Two sample sites with different access metrics, see text for explanation.

module is mostly transparent to the user; it will only display a startup dialog for initialization purposes, for example to change privacy settings. 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 and to track the user's movement (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). 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, 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.

B. 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 models relating resources (URLs) and usage patterns in physical space are expressed in terms of a metric based on *inertial ellipsoids*. The basic motivation is that of obtaining a smooth metric, where the spatial distribution of interest for a specific URL may have a preferred orientation in space.

The recommendation engine works on a set of s links, each identified by a unique id $l = 1, \dots, s$. Suppose that site l has been visited N_l times (possibly by different users), and let the set of points $P_i^l = (x_i^l, y_i^l)$, $1 \le i \le N_l$, 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 or to the length of time that a user spent on a web page. In the current version, for simplicity, all points are 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):

$$\bar{x}_l = \frac{1}{N_l} \sum_{i=1}^{N_l} x_i^l, \qquad \bar{y}_l = \frac{1}{N_l} \sum_{i=1}^{N_l} y_i^l,$$

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

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

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

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 $P = (x, y)$. The *preference* for a site l at point P is defined 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 $\pi^P = (\pi_1^P, \dots, \pi_s^P)$ of the site IDs having the property

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

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:

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

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 contains information about the status of a highway, and is mostly used by people driving along that road. Almost all accesses to the site have been performed along the highway. Because of the uni-dimensionality of the road, there is a strong correlation between the x and y coordinates of the points (the small black squares in the figure), and the resulting ellipse, 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 near 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 *a priori* information is built in the system, and the geographical relevance of a link is gradually inferred through the ellipsoid metric: every time a user clicks a link, the recommendation engine updates the database; inertial ellipsoids are periodically updated on the basis of the database records.

For a small number of database entries, a simple Euclidean distance model is appropriate, while the ellipsoid model, being characterized by more parameters, achieves a better representation when the number of samples increases to reach about one hundred points [13]. For the above reasons, the ellipsoid metric is chosen in the current work. Different metrics can be considered with appropriate modifications. In particular, complex shapes can be modeled by associating to a resource a *set* of ellipsoids.

The system is being currently tested as a one-page web application. The different building blocks shown in Figure 1 are implemented as separate C++ classes (the location broker and the recommendation index generator) and collected into one ActiveX component working also as position display (the top left map in Figure 3). The component, written in C++ with the Microsoft Foundation Classes library, interacts with the standard HTML form in the top frame of the browser to generate the recommendation page in the bottom frame.

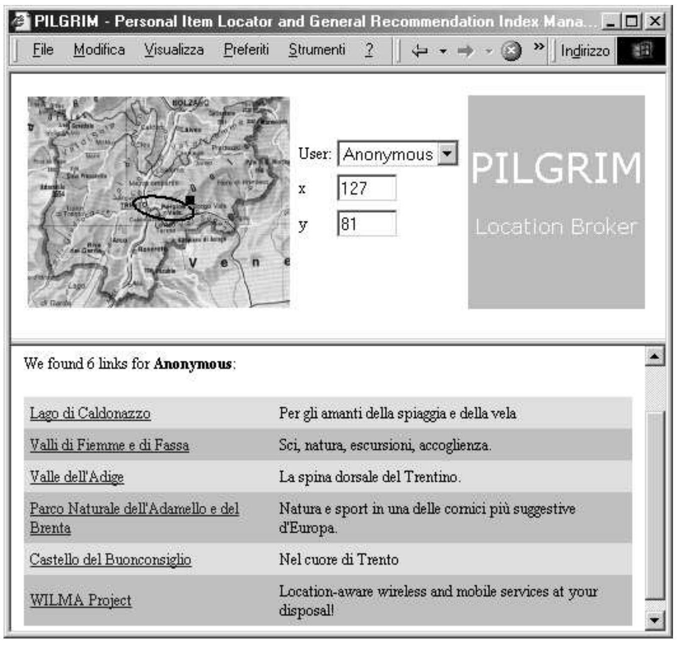
In order to have the system work with a small number of actual users, to avoid statistical fluctuations, the actual inertial ellipsoid is compensated by averaging with a fixed-radius circle having the same center. Let r be a default radius, for instance 1 kilometer; then

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

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



User: Anonymous

x 127

y 81

We found 6 links for **Anonymous**:

Lago di Caldonazzo	Per gli amanti della spiaggia e della vela
Valli di Fiemme e di Fassa	Sci, natura, escursioni, accoglienza.
Valle dell'Adige	La spina dorsale del Trentino.
Parco Naturale dell'Adamello e del Brenta	Natura e sport in una delle cornici più suggestive d'Europa.
Castello del Buonconsiglio	Nel cuore di Trento
WILMA Project	Location-aware wireless and mobile services at your disposal!

Fig. 3. Screenshot of the experimental PILGRIM system. User identity, location and map are shown on the top frame; the bottom frame contains an item recommendation list; the ellipse in the map represents the inertial ellipsoid of the link under the mouse pointer.

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 4 shows the behavior of the compensated inertial ellipsoid for a site being accessed by 100 users, one at a time. The graph shows the evolution of the ellipsoid when $r = 1000$m and $k = 500$: at the beginning, when the site is not yet very popular, the circular default prevails (though the aspect ratio of the graph shows it as an ellipse); later, the actual inertial ellipsoid outweighs the default circle and the correct shape is reached. Note that the very first estimate, only depending on the first access, can be off center with respect to the rest of the distribution.

IV. The ER-tree: a scalable implementation

The PILGRIM recommendation system relies on a site database recording the site name, URL, access statistics and preference ellipsoid. To execute queries about site location, some spatial indexes must be implemented. The number of sites is bound to grow rapidly as the system develops and learns, and the basic algorithm based on a sequential scans of all items in order to find the nearest neighbors rapidly becomes impractical. Smarter data structures, specifically tailored for speeding up spatial queries need to be implemented.

A wide range of spatial data structures has been proposed in the literature [19]. The data structure proposed in this paper is the ER-tree (*Elliptic R-tree*), an extension of the R-tree for the specific case of k-nearest-neighbor queries with different elliptic metrics.

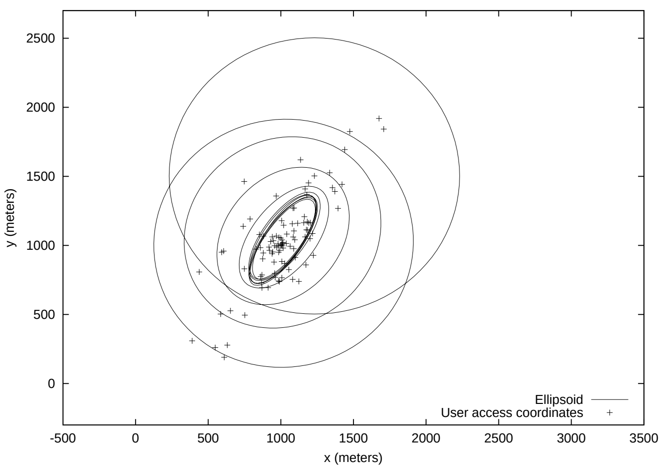


Fig. 4. Evolution in time of the compensated inertial ellipsoid representing the spatial distribution of users entering a site. Note that the initial estimate, the fixed-radius circle, is only based on the first access, and it can be off-center with respect to the overall distribution. Subsequent ellipses are shown every 10 steps.

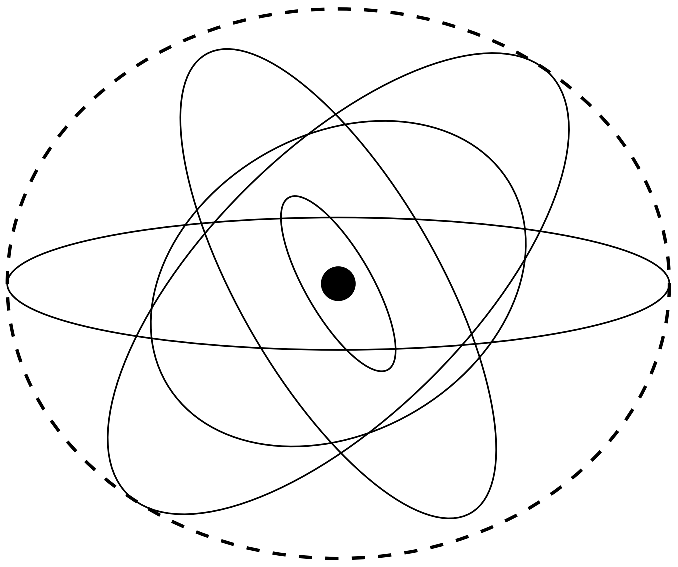


Fig. 5. Determining a lower-bound ellipsoid metric

Th proposed structure is based on the R-tree structure [20]. Let d be the number of spatial dimensions. The d-dimensional R-tree is a hierarchy of nested d-dimensional intervals. At each node, the corresponding interval is the bounding box of intervals in its subtree. Leaves are the objects (or pointers to the objects) in the database. Every node is required to have more than m and less than M sons, where m and M ($m \geq M/2$) are parameters to be adjusted; only the root is allowed to have as few as two sons. Algorithms to build the R-tree structure try to keep the tree in a good condition for queries by minimizing interval overlapping (which is in principle unavoidable). Point queries can be answered by restricting search only to branches whose bounding box actually enclose the point; nearest-neighbor queries can take advantage from branch-and-bound techniques. In fact, distance from a bounding box is a lower bound for distances from all enclosed objects.

The ER-tree data structure has been implemented in this work for $d = 2$. The indexed objects in the database are the centroids of the ellipses, having pointwise extension. In the considered case, however, Euclidean distance from a bounding box is not a lower bound for the actual distances of the enclosed objects. In fact, in the aforementioned site model every site has a different elliptical metric. To make the lower bound work, every time a bounding box is calculated from the enclosed objects, a corresponding elliptical metric must be devised and applied.

Figure 5 shows how the overall metric for a bounding box is calculated. All ellipses for the enclosed objects are translated to the origin, and a “bounding ellipse” is built such that its major axis corresponds with the largest major axis of the ellipses, while the minor axis is calibrated in order to contain all ellipses. Since every ellipse represents the unit-distance locus for the corresponding metric, it is straightforward that the overall metric is a lower bound for all enclosed metrics. Every node in the tree contains, as an additional information, the overall metric corresponding to its bounding box.

Figure 6 shows part of an ER-tree, both by its actual spatial structure and by its tree representation. Node **a** is shown with its corresponding bounding box (the dashed rectangle) and its subnodes (**b** to **h**). All subnodes, with the exception of **d** and **g**, contain objects, i.e. leaves of the ER-tree; nodes **d** and **g** contain lower-level nodes, which will be further decomposed. The thick line surrounding the bounding box of **a** is the locus of points having unit distance from the rectangle, and is determined by the overall metric of the node.

V. Experimental evaluation

To evaluate the performance of the data structure with respect to k-nearest-neighbor queries, some tests have been performed on randomly-generated sites. On a square one-kilometer area a certain number N (100 to 100000) of sites are uniformly spread. Every site is assigned an ellipse. The major axes of the ellipses follow a Gaussian distribution with given average μ and standard deviation σ. The inclination of the major axis and the eccentricity are uniformly distributed, the first in the interval $[-\pi/2, \pi/2)$, the second in $[e_{\text{min}}, 1]$, where e_{min} is the smallest admitted eccentricity. We used a distribution of $N = 100$ sites with $e_{\text{min}} = 0.1$ and $\mu = \sigma = 100$m.

The ER-tree depends on one parameter M, the node maximum degree or “node size”, while the corresponding minimum for non-root nodes has been set to $m = \lfloor M/2 \rfloor$. The nearest neighbors search algorithm depends on the number k of neighbors to be returned.

Figure 7 shows the performance comparison between a linear search in an array of sites and the branch-and-bound search on the ER-tree for different distributions. Along the x axis the number of sites is reported; the y axis represents the average performance improvement obtained by the ER-tree search. Performance is tested by counting the number of distance evaluations. In fact, from preliminary experiments

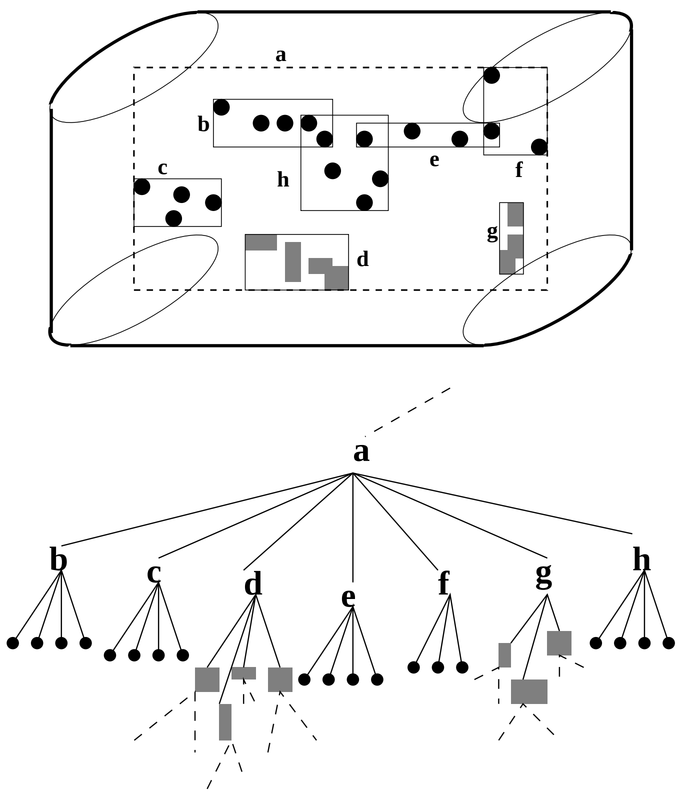


Fig. 6. An ER-tree node (top) and corresponding ER-tree representation (bottom).

TABLE I
COMPARISON BETWEEN LINEAR ARRAY AND ER-TREE SEARCH FOR A 100000-SITE SAMPLE

	R-tree	Array	Ratio	% Differ.
Time (ms)	5.00	42.00	0.119	88.1%
Distance evals	9672.8	100000	0.097	90.3%
Memory (bytes)	22073344	21600000	1.022	2.2%

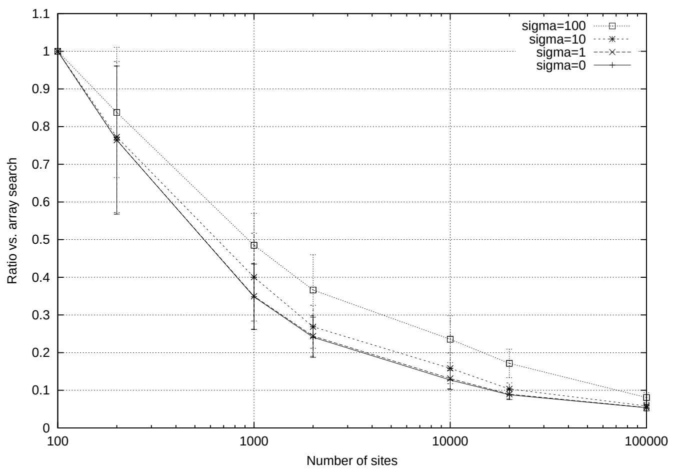


Fig. 7. Comparison between linear array and ER-tree search for different item populations and different radius distributions.

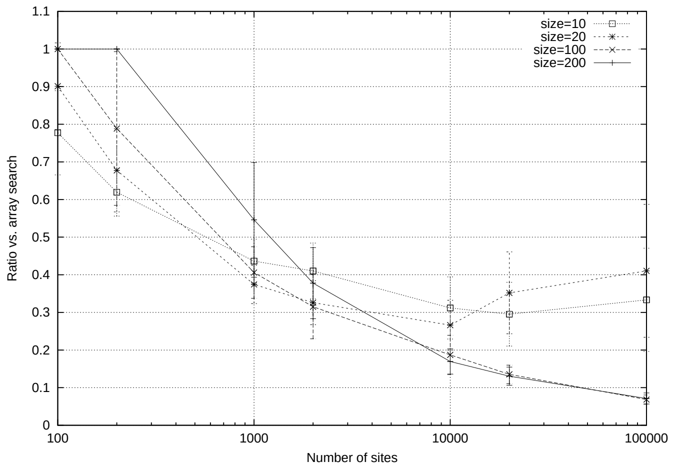
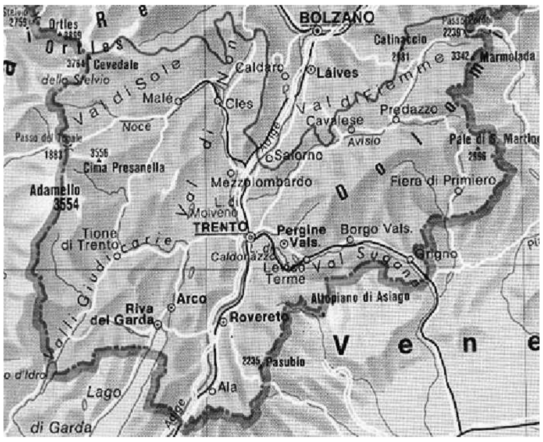


Fig. 8. Comparison between linear array and ER-tree search for different item populations and different node sizes.

such as that shown in Table I we see that counting the number of evaluations gives a good measure of the actual time improvement, while time itself, when tested on a time-shared machine, can suffer from many external influences. Parameters of the test are $k = 10$, $\mu = 100$m, $e_{\min} = 1$ (metrics are isotropic), $M = 100$. The different lines in the graph show the behavior for different values of σ. In particular for $\sigma = 0$ all sites have exactly the same radius. In this case, all metrics are equivalent and the lower bound estimation on distances is more effective. Every point in the graph shows the average of 100 tests, and error bars indicate the 95% confidence interval. If radii are allowed to differ, up to a $\sigma = 100$m standard deviation, then performance slightly degrades. Note that for $N = 100$ sites no improvements are obtained, because if $N \leq M$ only the root is filled and a sequential search over all sites is therefore performed. However, improvements up to 90% are obtained as the number of items grows t $N = 100000$.

Figure 8 shows the same type of comparisons, where the standard deviation of the major axis is $\sigma = 100$m, eccentricity varies from a minimum of $e_{\min} = .1$ and the different lines show how the performance changes if the ER-tree node size varies from $M = 10$ to $M = 200$. Note that for small node sizes the advantage over sequential search is almost negligible (if $N \leq M$ all nodes must be visited anyway). Small node sizes become unpractical for a large number of nodes, probably because the depth of the tree grows considerably, and a long downwards exploration must be undertaken before arriving to the leaves.

A second set of experiments has been performed in order to evaluate the performance of the system in a more realistic context. The population structure of the Trentino region, shown in the upper part of Figure 9, has bee used in order to generate a more structured pattern of web sites. Trentino is a mountain region with most people living within short distance from the bottom of the valleys and concentrated near the main towns. The system is likely to be used also by people moving along roads, or by tourists spread along the whole territory. A simulated usage test with $N = 100$ web sites resulted in the pattern shown in the bottom part of Figure 9. Most ellipsoids



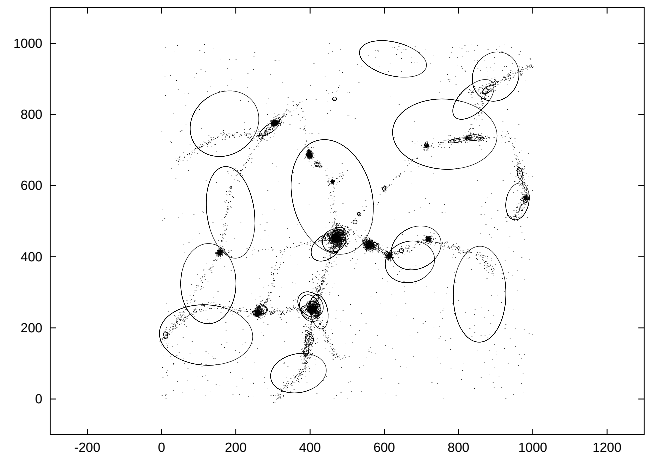


Fig. 9. The Trentino region (above) and ellipsoids of interest for localized web sites (below: small dots represent distribution of single users; while many are concentrated in towns and valleys, some are spread throughout the territory).

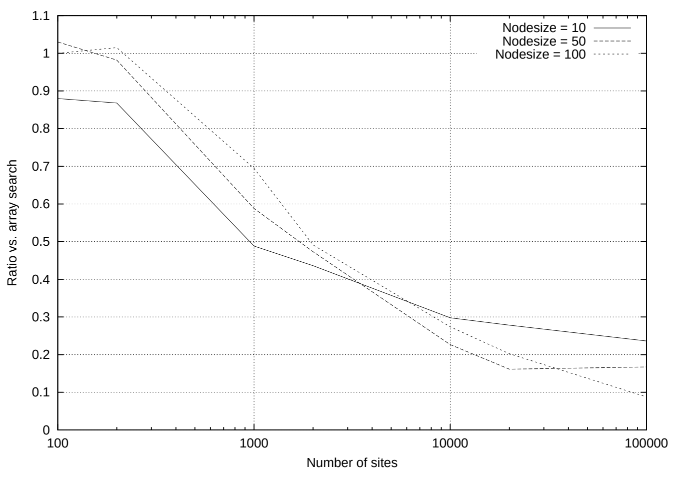


Fig. 10. Performance ratio of the R-tree data structure versus linear scan for different node sizes and different numbers of indexed sites.

are concentrated in towns, but a few are of wide interest: they may correspond to a famous resource or to a mountain visible from a large distance.

Figure 10 shows some preliminary comparisons between linear scans and ER-Trees for the simulated environment. Every point represents the average of 100 ten-nearest-neighbor searches, and the error bars show the 95% confidence interval for the mean. Note that the improvement with respect to linear scan is comparable to that obtained on the random distribution in Figures 7 and Figure 8.

VI. Discussion and conclusions

We proposed a location-aware recommender system that is integrated with a basic browser and that filters resources (URLs) for a specific user. The filter is adaptive and takes into account both the *current user location* and models relating resources to geographic locations built by mining the previous history of usage. The main advantage with respect to existing systems lies in the automated creation of models (without an explicit design to couple locations and URLs), the flexibility and the independence from ad-hoc systems realized by resource owners (that naturally tend to be biased, especially for mobile commerce related sites).

To allow a large degree of scalability demanded by a large scale implementation, a hierarchical data structure has been proposed and analyzed. The results on a model that considers the population and building distribution of our region (Trentino) show that performance of the ER-tree structure does not degrade when implemented on a more structured pattern of usage.

Techniques to select the appropriate privacy level can be easily implemented. For example, if the user feels uneasy about communicating her own precise 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 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.

A future issue on the agenda is the implementation as a distributed system where local databases contain information about items close to the server location, and a peer-to-peer content distribution scheme enables synchronization among all local servers. In addition, the integration of the location-broker with traditional recommendation system will permit specialized implementations (like for example a system dedicated to gourmet restaurants, to tourists interested in art and monuments, to mobile shopping, etc.).

VII. Acknowledgments

References

[1] J. Steinberg and J. Pasquale, "A web middleware architecture for dynamic customization of content for wireless clients," in *Proceedings of Eleventh International World Wide Web Conference - WWW2002*, (Honolulu, Hawaii, uSA), May 2002.

[2] S. Duri, A. Cole, J. Munson, and J. Christensen, "An approach to providing a seamless end-user experience for location-aware applications," in *Proceedings of the fi rst international workshop on Mobile commerce*, (Rome), pp. 20–25, July 2001.

[3] T. Bohnenberger, A. Jameson, A. Krüger, and A. Butz, "User acceptance of a decision-theoretic, location-aware shopping guide," in *IUI 2002: International Conference on Intelligent User Interfaces* (Y. Gil and D. B. Leake, eds.), pp. 178–179, New York: ACM, 2002. Available from http://dfki.de/∼jameson/abs/BohnenbergerJK+02.html.

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

[5] P. Resnik and H. R. Varian, "Recommender systems," *Communications of the ACM, special issue*, vol. 40, pp. 56–58, Mar. 1997.

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

[7] M. Balabanovič and Y. Shoham, "Fab: Content-based, collaborative recommendation," *Communications of the ACM*, vol. 40, pp. 66–72, Mar. 1997.

[8] 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*, vol. 40, pp. 77–87, Mar. 1997.

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

[10] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of WWW10*, May 2001.

[11] M. O'Connor and J. Herlocker, "Clustering items for collaborative filtering," tech. rep., University of Minnesota, department of Computer Science, Minneapolis, USA, 2000.

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

[13] M. Brunato and R. Battiti, "PILGRIM: A location broker and mobility-aware recommendation system," Tech. Rep. DIT-02-0092, Dipartimento di Informatica e Telecomunicazioni, Universit`a di Trento, Oct. 2002. submitted.

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

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

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

[17] 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*, vol. 9, July 2002.

[18] R. Battiti, M. Brunato, and A. Villani, "Statistical learning theory for location fingerprinting in wireless LANs," Tech. Rep. DIT-02-0086, Dipartimento di Informatica e Telecomunicazioni, Universit`a di Trento, Oct. 2002.

[19] V. Gaede and O. Günther, "Multidimensional access methods," tech. rep., Institut für Wirtschaftsinformatik, Humboldt-Universität zu Berlin, 1996.

[20] A. Guttman, "R-tree: A dynamic index structure for spatial searching," in *Proceedings of ACM SIGMOD*, 1984.