Time Aware Web Users Clustering

Sophia G. Petridou, Vassiliki A. Koutsonikola, Athena I. Vakali, and Georgios I. Papadimitriou, *Senior Member, IEEE*

***Abstract*—Web users clustering is a crucial task for mining information related to users needs and preferences. Up to now, popular clustering approaches build clusters based on usage patterns derived from users' page preferences. This paper emphasizes the need to discover similarities in users' accessing behavior with respect to the time locality of their navigational acts. In this context, we present two time aware clustering approaches for tuning and binding the page and time visiting criteria. The two tracks of the proposed algorithms define clusters with users that show similar visiting behavior at the same time period, by varying the priority given to page or time visiting. The proposed algorithms are evaluated using both synthetic and real datasets and the experimentation has shown that the new clustering schemes result in enriched clusters compared to those created by the conventional non-time aware users clustering approaches. These clusters contain users exhibiting similar access behavior not only in terms of their page preferences but also of their access time.**

***Index Terms*—Web mining, Web users clustering, Navigation, Access time.**

I. Introduction

DESPITE Web's remarkable adoption, Web users often experience problems of low precision or irrelevance in their searching, low accessing speeds (due to information overload) and outdated or out of their interests personalized information. Thus, from the Web information providers perspective, it is important to organize their data and to address their users according to their preferences and needs. Web users clustering so far has proposed techniques to organize users into clusters based on their navigational behavior, i.e. visiting patterns are identified and compared in order to assign users to the same or different clusters. The task of Web users' clustering is crucial and has been studied in various application frameworks, since for example based on users clusters, Web-based companies may provide dynamic content (advertisements, offers, customized guides) and decide their market strategies and administrators may restructure or redesign their sites and improve their performance (by user-tailored caching and prefetching policies) etc.

Earlier research efforts have focused on grouping users that present similar page preferences mainly identified by Web server log files which explicitly record the browsing behavior of each user. Such efforts have used various clustering approaches, which are based on identifying common patterns in users navigational behavior. It should be noted that these methods are either *model-based* or *similarity-based*. The Expectation-Maximization (EM) algorithm is the most popular model-based approach. It is typically used to provide associations among users and pages [1], [2] as well as to identify user profiles [3]. The most popular similarity-based algorithm is K-means [4] which has been used with popular distance measures such as the squared Euclidean, the cosine and the Manhattan distances [5], [6], [7]. Among other approaches, hierarchical and partitional [8] clustering techniques are used to evaluate similarity of users' sessions [6] while a two-phase clustering is used to perform pattern analysis and classification based on users' registration information [9].

Time-related issues in Web users' clustering have been addressed in previous work by mainly considering the duration of a user's accessing on a page and the succession of their visits (the so-called "clickstreams") [10], [11]. However, such time consideration does not clearly identify users needs since the time spent by a user in a particular page is not always an actual indication of the user's preferences (e.g. users often leave their browsers idle during breaks, meetings or in order to start browsing in a new window) whereas capturing similar "clickstreams" is done in an overall time span and not on simultaneous intervals. An oversimplified approach [12] uses a bit value to identify the access time (i.e. 0 and 1 indicates day or night visit) but does not capture more detailed time preferences of users' visits. In summary, it is important to emphasize that the common ground of related work is the effort to identify common page visiting pathways irrelevant to the actual time when these occurred, even in cases where the notion of time is used.

This work is inspired by the fact that in the framework of several current applications, the time locality of page visiting patterns needs to be also considered, since in reality users exhibit varying accessing behavior at different times (e.g. on a yearly, monthly or even daily scale). Therefore, the consideration of access time along with the page preferences is imperative in clustering since, in fact, the time in which users perform certain page visits is a crucial criterion for characterizing their particular needs and preferences. We may identify numerous scenarios in which time is critical in the Web users clustering process, since time along with page visiting is important in:

- *commercial Web-based applications*, e.g., some users tend to buy gifts, cards or books only during Christmas while others make travel reservations only in summer, therefore the fact that two users have visited the particular products site is not enough to identify the customers similarity in browsing preferences. This is crucial since at a certain time of the year the company might organize a particular promotion campaign.
- *personalization and/or recommendation engines*, e.g. in a recommendation engine the clustering of users showing

S. Petridou, V. Koutsonikola, A. Vakali and G. Papadimitriou are with the Department of Informatics, Aristotle University of Thessaloniki, Greece, e-mail: {spetrido,vkoutson,avakali,gp}@csd.auth.gr.

similar navigational behavior over the same time span is important, since their personalization information or the recommendations proposed will be tailored accordingly.
- *e-learning environments*, e.g. in a University Department site, just the fact that two users have visited the Labs same pages is not indicative by its own for assigning them to the same cluster, since visits during the summer months would probably correspond to students interested in enrolling to a course (some of them might never visit the page again), while autumn months visits would probably refer to students enrolled to courses.
- *low-level applications*, e.g. identifying groups of users exhibiting similar navigational behavior in terms of their page preferences and access time may be beneficial for applications like Web caching and prefetching which affect the Web server performance.

Based on the above, in this paper we highlight the fact that grouping Web users based on their navigational behavior should be faced as a twofold problem that will: (i) deal with the different users' page preferences and (ii) identify the time dependencies involved in the usage navigational patterns. Thus, the problem that has to be addressed should combine the above two criteria namely the users' page preferences i.e. the page aspect and the time their visits were logged i.e. the time aspect. Since the proposed approach aims at advancing the earlier ones (which considered only the page preferences), it is crucial to determine the way that the time aspect will be incorporated in the clustering process.

Therefore, two tracks of problems are identified, where in the first one the role and importance of time is tuned with page preferences while in the second one the time and page aspects are bound. In the tuning problem it is important to specify the aspect (i.e. the page or time) that is initially considered since this aspect will guide the clustering process. Thus, we adopt two algorithmic approaches that differ in their initialization step and tune the two aspects based on a weight factor. The first tuning approach initiates with the page preferences and then proceeds to the time aspect while the second one follows the reverse logic. The binding problem captures the two aspects based on a structure that incorporates page preferences and access time simultaneously.

The remainder of this paper is organized as follows. Section II focuses on the structures and distance measures used to represent users and capture their similarities respectively. Section III describes the two tracks of problems defined to tune and bind page and time aspects, whereas Section IV presents the algorithms and discusses their complexity. Section V presents the experimentation carried out in both synthetic and real data and highlights their implementation perspective. Finally, conclusions and future work insights are given in Section VI.

II. Capturing users' access behavior

We consider a particular Web usage framework where we have (as a source) log files which capture the users' navigational behavior. Moreover, we define the notion of the timeframe which refers to the particular time basis (i.e. period) on which we examine users' actions. For example, working with a monthly log file we can define the timeframe as one day while in an annual log file one month could serve as the respective timeframe. These choices are based on the log file's time period, however, in the proposed approaches, timeframe's definition can be adjusted according to the demands imposed by the particular application framework on which we work on. Therefore, we have n, p and t to denote the number of users, pages and timeframes respectively while U denotes the users' set $U = \{u_1, \ldots, u_n\}$. Notation summary is given in Table I.

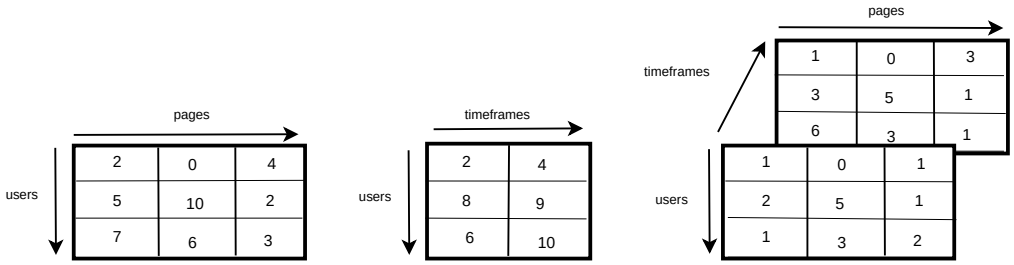
TABLE I
Basic Symbols Notation.

Symbol	Definition
n, p, t	Number of users, pages and timeframes
U	Users' set $\{u_1, \ldots, u_n\}$
PV	The n x p users' page visiting table
TV	The n x t users' time visiting table
PTV	The n x p x t users' page-over-time visiting table
d_p	Users' distance over pages
d_t	Users' distance over timeframes
d_{pt}	Page-over-time users' distance

As emphasized in [13], prior to clustering, appropriate structures or patterns choice, largely affects the accuracy and quality of the yielded clusters. Therefore, we need to define appropriate structures to represent users, since based on these structures we may then define their similarities and end up to simple and easily interpretable clusters. In general, patterns can be represented by using various structures, such as strings or trees [14], whereas the vector-space model has been extensively used [15] since vectors values can be either quantitative (continuous values: e.g. weight, discrete values: e.g. the number of visits of a Web user or interval values: e.g. the duration of an event) or qualitative (nominal: e.g. "red" or ordinal: e.g. "cool") [16]. Since in our web usage approach we have discrete values, the proposed time-aware clustering uses the ideas of vector-space model for users' representation.

More specifically, we define three different user visiting structures in order to capture all aspects of interrelations in page and time visiting. A vector is used to represent the frequency of a user's visits to particular pages (with no information about the time of visits) while a second one records the frequency of the user's visits at particular timeframes (with no information about which page was visited). Moreover, the lack of the complementary information in each of these vectors, motivated us to define a table which will incorporate the overall information (seen as a set of vectors). In particular, this table represents the frequency of visits to particular pages incorporating the exact knowledge about the timeframes of these visits. These structures are summarized next:

1) **page visiting vectors:** A page visiting vector $PV(i,:)$, where $i = 1, \ldots, n$, represents a user's accessing behavior with respect to page visiting solely. It is a multivariate



(a) n x p page visiting table *PV* (b) n x t time visiting table *TV* (c) n x p x t page-over time visiting table *PTV*

Fig. 1. Page and time users' visiting structures.

vector consisting of p measurements:

$$PV(\mathbf{i},:) = (PV(i,1),\ldots,PV(i,p))$$

where the $PV(i,j)$ element, $j = 1,\ldots,p$, indicates the number of times the user i visits the page j. All the $PV(i,:)$ vectors are then organized in the two dimensional n x p users' page visiting table PV. For example, in Fig. 1(a), which depicts the table *PV*, the fact that $PV(2,2) = 10$ means that the user identified as 2 has made 10 visits to the page 2.

2) **time visiting vectors:** A time visiting vector $TV(i,:)$, where $i = 1,\ldots,n$, represents a user's accessing behavior with respect to time (timeframes). It is also a multivariate vector consisting of t measurements:

$$TV(\mathbf{i},:) = (TV(i,1),\ldots,TV(i,t))$$

where the $TV(i,l)$ element, $l = 1,\ldots,t$, indicates the number of times the user i visits the whole site (all the p pages) during the l timeframe. All the $TV(i,:)$ vectors are organized in the two dimensional n x t users' time visiting table *TV*. For example, in Fig. 1(b), which depicts the table *TV*, the fact that $TV(2,2) = 9$ means that the user identified as 2 has made 9 visits to the whole site during the timeframe 2.

3) **page-over-time visiting tables:** A page-over-time visiting table $PTV(i,:,:)$, where $i = 1,\ldots,n$ also represents a user's accessing behavior but with respect both to page and time aspects. It consists of p x t measurements:

$$PTV(\mathbf{i},:,:) = \begin{pmatrix} PTV(i,1,1) & \ldots & PTV(i,1,t) \\ PTV(i,2,1) & \ldots & PTV(i,2,t) \\ \vdots & \vdots & \vdots \\ PTV(i,p,1) & \ldots & PTV(i,p,t) \end{pmatrix}$$

where the $PTV(i,j,l)$ element, $j = 1,\ldots,p$ and $l = 1,\ldots,t$, indicates the number of times the user i visits the page j during the l timeframe while the $PTV(i,j,:)$ denotes the timeframes vector of the user i over the page j (jth row of $PTV(i,:,:)$ table). All the $PTV(i,:,:)$ tables are organized in the three dimensional n x p x t users' page-over-time visiting table *PTV*. In Fig. 1(c), the fact that $PTV(2,2,1) = 5$ means that the user identified as 2 has made 5 visits to the page 2 during the timeframe 1.

Based on the above, it is obvious that the elements of the page visiting vectors $PV(i,:)$, the time visiting vectors $TV(i,:)$ and the page-over-time visiting tables $PTV(i,:,:)$ are related. Following the previous examples, it holds that $\sum_{l=1}^{t} PTV(2,2,l) = PV(2,2) = 10$, i.e. the number of visits of a user to a page is split over the underlying timeframes in order to capture the exact timeframes of these visits. Respectively, $\sum_{j=1}^{p} PTV(2,j,2) = TV(2,2) = 9$, i.e. the number of visits of a user during a timeframe is split to capture the exact visited pages. (1) gives the relationship between the *PV* and *PTV* tables, where $i = 1,\ldots,n$, $j = 1,\ldots,p$ and $l = 1,\ldots,t$:

$$PV(i,j) = \sum_{l=1}^{t} PTV(i,j,l) \tag{1}$$

while (2) gives the relationship between the *TV* and *PTV* tables:

$$TV(i,l) = \sum_{j=1}^{p} PTV(i,j,l) \tag{2}$$

Furthermore, given that in the above two dimensional structures each element captures visiting frequencies, it is clear that the overall number of visits of a particular user to a specific site is embedded in both of these tables. Therefore, it holds that $\sum_{j=1}^{p} PV(2,j) = \sum_{l=1}^{t} TV(2,l)$. Given the user i, where $i = 1,\ldots,n$, (3) gives the relationship between the *PV* and *TV* tables, where $j = 1,\ldots,p$ and $l = 1,\ldots,t$:

$$\sum_{j=1}^{p} PV(i,j) \;= \sum_{l=1}^{t} TV(i,l) \tag{3}$$

In summary, given that users' accessing behavior depends on the time of their navigation, our users' representation captures both their page preferences, by the page visiting vectors $PV(i,:)$, and their time locality by the time visiting vectors $TV(i,:)$. At the same time, the page-over-time visiting tables $PTV(i,:,:)$ captures simultaneously the page and time aspect of users' visits.

A. Calculating users' distance

Devising appropriate distance measures is fundamental in a clustering process, and so far it is quite common to evaluate dissimilarity between two patterns by using a distance measure

in order to find if they are relevant or not [13]. To proceed with our Web users' clustering process, we employ the Squared Euclidean distance[1] which is a well-known and widely used distance measure in the vector-space model [5],[6],[7]. Then, the evaluation of dissimilarity between two users may be expressed by their distance that can be based either on their visiting vectors (page or time) or on their visiting table (page-over-time). So, we define three types of distances:

1) **users' distance over pages:** When only the pages preferences are taken into consideration, the distance between two users must be calculated over each of the p pages. In this case, we will use the expression $d_p(u_x, u_y)$, where $u_x, u_y \in U$ to denote the distance between the page visiting vectors $PV(x,:)$ and $PV(y,:)$ of the users u_x and u_y. Their Squared Euclidean distance is:

$$d_p(u_x, u_y) = \|PV(x,:) - PV(y,:)\|^2$$

Example. Consider the users identified as 1 and 3 in Fig. 1(a). Their distance over pages will be $d_p(u_1, u_3) = \|PV(1,:) - PV(3,:)\|^2 = \|(2,0,4) - (7,6,3)\|^2 = 62$.□

2) **users' distance over timeframes:** When only the time locality of visits is taken into consideration, the distance between two users must be calculated over each of the t timeframes. In this case, we will use the expression $d_t(u_x, u_y)$ to denote the distance between the time visiting vectors $TV(x,:)$ and $TV(y,:)$ of the users u_x and u_y. Their Squared Euclidean distance is:

$$d_t(u_x, u_y) = \|TV(x,:) - TV(y,:)\|^2$$

Example. Considering the same users (1 and 3) in Fig. 1(b), their distance over timeframes will be $d_t(u_1, u_3) = \|TV(1,:) - TV(3,:)\|^2 = \|(2,4) - (6,10)\|^2 = 52$. □

3) **page-over-time users' distance:** When both pages preferences and their time locality are taken into consideration, the distance between two users must be calculated over each of the p pages separately. In this way, we can capture the dissimilarity between users over each page for all timeframes. As a consequence, two users' access behaviors are considered to be similar not only if they visit the same pages with similar frequency but also if they visit the same pages at the same timeframes with similar frequency. So, we define the distance of page-over-time visiting tables to be the sum of distances over all pages.
We will use the expression $d_{pt}(u_x, u_y)$ to denote the distance between the page-over-time visiting tables $PTV(x,:,:)$ and $PTV(y,:,:)$ of the users u_x and u_y. Since each row of these p x t tables is a vector (i.e. $PTV(x,j,:)$ and $PTV(y,j,:)$ respectively), we compute their distance by calculating the sum of the Squared Euclidean distances between the corresponding p rows (i.e. vectors) of the tables:

$$d_{pt}(u_x, u_y) = \sum_{j=1}^{p} \|PTV(x,j,:) - PTV(y,j,:)\|^2$$

Example. The page-over-time distance of the users identified as 1 and 3 in Fig. 1(c) will be $d_{pt}(u_1, u_3) = \sum_{j=1}^{3} \|PTV(1,j,:) - PTV(3,j,:)\|^2 = \|(1,1) - (1,6)\|^2 + \|(0,0) - (3,3)\|^2 + \|(1,3) - (2,1)\|^2 = 48$.□

[1]The Squared Euclidean distance uses the same equation as the Euclidean distance, but does not take the square root. For two points $P = (p_1, p_2, \cdots, p_n)$ and $Q = (q_1, q_2, \cdots, q_n)$ in n-space their Squared Euclidean distance is defined as: $\|p_i - q_i\|^2$

III. Problem Formulation

TABLE II
Clustering and Objective Functions.

Symbol	Definition
CL	Clustering process
k	Number of clusters
U_j	Cluster, $j = 1, \ldots, k$
c_j	Cluster representative, $j = 1, \ldots, k$
$f(u_i, U_j)$	Function membership of user u_i to cluster U_j
CP	k x p cluster representative's page visiting table
CT	k x t cluster representative's time visiting table
CPT	k x p x t cluster representative's page-over-time visiting table
E_p, E_t	Page and time objective function
E	Tuning objective function
E_{pt}	Binding objective function

In the proposed clustering, it is important to identify the type of problem to be solved, since as mentioned earlier, we are dealing with a twofold problem which at the same time involves two distinct criteria: the users' page preferences and the timeframe of their occurrence. We consider that under the proposed CL time-aware clustering process, k denotes the number of clusters and U is the set of users $U = \{u_1, \ldots, u_n\}$ to be clustered. Then, $U_1, \ldots, U_k$ denote each of the k clusters consisting of $|U_1|, \ldots, |U_k|$ members respectively. Under this notation, the underline clustering process CL is defined as the assignment of users to k users' groups (i.e. clusters):

$$CL : \{1, \ldots, n\} \longrightarrow \{1, \ldots, k\}$$

such that the users assigned to each cluster are more similar to each other than to the users assigned to different clusters on the basis of: (i) their page preferences (ii) the timeframe these preferences were logged.

Membership of a user u_i, where $i = 1, \ldots, n$, to a cluster U_j, where $j = 1, \ldots, k$, is defined by the function f as follows:

$$f(u_i, U_j) = \begin{cases} 1 & \textit{if } u_i \in U_j \\ 0 & \textit{otherwise} \end{cases}$$

A. *The* Tuning Time-aware Clustering *problem*

Let us consider an arbitrary cluster U_j, where $j = 1, \ldots, k$, of the users' set U in the n x p space. The representation of the cluster U_j when a clustering process CL is applied to it, collapses the users belonging to U_j into a single point (i.e. the

mean value which does not correspond to an existing user). We call this point cluster's representative c_j (also known as centroid) as each user $u_i \in U_j$ is represented by c_j. Given the page visiting vectors of $u_i \in U_j$, we can define the page visiting vector of c_j in the n x p space as follows:

$$\boldsymbol{CP}(j,:) = \frac{\sum_{i=1}^{n} f(u_i, U_j) * \boldsymbol{PV}(i,:)}{|U_j|}$$

Since both $PV(i,:)$, where $i = 1, \ldots n$, and $CP(j,:)$, where $j = 1, \ldots k$, are vectors, their dissimilarity is measured by their page visiting distance $d_p(u_i, c_j)$. Considering all clusters, we define the *page objective function* E_p to be the sum of distances over pages between each user and the representative of the cluster that the user is assigned to.

$$E_p = \sum_{j=1}^{k} \sum_{u_i \in U_j} d_p(u_i, c_j)$$

Considering the cluster U_j in the n x t space, we can respectively define its c_j representative based on time visiting vectors of $u_i \in U_j$ as:

$$\boldsymbol{CT}(j,:) = \frac{\sum_{i=1}^{n} f(u_i, U_j) * \boldsymbol{TV}(i,:)}{|U_j|}$$

In this case, the distance between c_j and u_i is measured by their time visiting distance $d_t(u_i, c_j)$ while the *time objective function* E_t computes the sum of distances over timeframes between each user and the representative of the cluster that the user is assigned to.

$$E_t = \sum_{j=1}^{k} \sum_{u_i \in U_j} d_t(u_i, c_j)$$

Tuning page and time objective functions can be treated as a multi-objective optimization problem. Such problems have been studied in many different domains [17], [18], [19] and a crucial difficulty in optimizing a multi-objective problem is that no single optimal solution exists. Instead, an optimal solution exists for each objective function in the solution space. Moreover, finding an optimal solution for one objective function may require accepting a poor solution for the others. Therefore, in our case, a clustering solution that can optimize the page aspect will most probably be non-optimal according to the time aspect. To disambiguate the multi-objective problem's solution, we assign different weights to each objective function in order to consider them in conjunction. We define the *tuning objective function* E to capture the properties and quality of the desired clustering solution and formulate the clustering problem. This objective function will guide our clustering and is defined as:

$$E = \alpha * E_p + (1 - \alpha) * E_t \tag{4}$$

where E_p and E_t are the objective functions focusing on page and time aspect respectively, and α is the weight factor with values in [0..1]. Then, at the one end, when $\alpha = 1$, $E = E_p$, i.e. our solution proposes an assignment based only on users' page preferences and completely discards the time aspect. At the other end, when $\alpha = 0$, $E = E_t$ and the solution is based only on time locality of users' visits and does not take into account their page preferences. For any other value of α the clustering solution considers both criteria at balanced weights.

Based on the above, we define the TUNING TIME-AWARE CLUSTERING problem as follows:

*Problem 1 (*TUNING TIME-AWARE CLUSTERING): Given a set U of n users in both n x p and n x t space, an integer value k, and the tuning objective function E, find a CL clustering of U into k clusters such that the E is minimized.■

1) Distance normalization: Our TUNING TIME-AWARE CLUSTERING problem aims at minimizing the tuning objective function E defined in (4). However, the formulation of (4) cannot handle objective functions that change in different scales, because a weighted sum of them would be meaningless. One way to overcome this difficulty is to normalize the distance values.

Let $d_p^*(u_i, c_j)$ and $d_t^*(u_i, c_j)$ be the normalized values of $d_p(u_i, c_j)$ and $d_t(u_i, c_j)$ respectively. It will be proved that $d_p^*(u_i, c_j)$ and $d_t^*(u_i, c_j)$ are of the same scale.

Given that $d_p(u_x, u_y)$ is the Squared Euclidean distance between the visiting vectors $PV(x,:), PV(y,:)$ of the users u_x and u_y, where $u_x, u_y \in U$, we denote $maxd_p$ to be the maximum distance between all pairs of u_x and u_y. Thus, $0 \leq d_p(u_x, u_y) \leq maxd_p$.

Defining the normalized distance $d_p^*(u_x, u_y)$ to be:

$$d_p^*(u_x, u_y) = \frac{d_p(u_x, u_y)}{maxd_p}$$

we conclude that:

$$0 \leq d_p^*(u_x, u_y) \leq 1$$

Based on the above, the definitions of E_p and E_t are updated as follows:

$$E_p = \sum_{j=1}^{k} \sum_{u_i \in U_j} d_p^*(u_i, c_j) \tag{5}$$

and

$$E_t = \sum_{j=1}^{k} \sum_{u_i \in U_j} d_t^*(u_i, c_j) \tag{6}$$

Lemma 1: Let U_j, $1 \leq j \leq k$, be the jth of the k clusters of a users' set U after a CL clustering process, $\boldsymbol{CP}(j,:)$ be cluster representative's page vector, $|U_j|$ be the number of its members and h, g denote the 1st and $|U_j|$th member of the U_j cluster respectively. Given that the page objective function E_p is the sum of the normalized distances between users and their cluster representative, it holds that:

$$0 \leq E_p \leq n \tag{7}$$

Proof: Given the (5), we must prove that for every user $u_i \in U_j$, $0 \leq d_p^*(u_i, c_j) \leq 1$, $1 \leq i \leq n$. By the definition of normalized distances it holds that $d_p^*(u_i, c_j) = \frac{d_p(u_i, c_j)}{maxd_p}$. So,

we shall prove that $d_p(u_i, c_j) \leq maxd_p$.

$$
\begin{aligned}
d_p(u_i,c_j) &= \|PV(i,:) - CP(j,:)\|^2 \\
&= \sqrt{|PV(i,:) - (\frac{PV(h,:)+\ldots+PV(g,:)}{|U_j|})|^2}^2 \\
&= |PV(i,:) - \frac{PV(h,:)}{|U_j|} - \ldots - \frac{PV(g,:)}{|U_j|}|^2 \\
&= \frac{1}{|U_j|^2}||U_j|PV(i,:) - PV(h,:) - \ldots - PV(g,:)|^2 \\
&= \frac{1}{|U_j|^2}|PV(i,:) - PV(h,:) + \ldots + PV(i,:) - PV(g,:)|^2 \\
&\overset{a \leq \sqrt{a^2}}{\leq} \frac{1}{|U_j|^2}|\sqrt{(PV(i,:) - PV(h,:))^2} + \ldots \\
&\qquad \ldots + \sqrt{(PV(i,:) - PV(g,:))^2}|^2 \\
&= \frac{1}{|U_j|^2}|Euclidean(u_i,u_h) + \ldots + Euclidean(u_i,u_g)|^2 \\
&\leq \frac{1}{|U_j|^2}|U_j|^2 |maxEuclidean|^2 \\
&= maxd_p
\end{aligned}
$$

■

Lemma 2: Let U_j, $1 \leq j \leq k$, be the jth of the k clusters of a users' set U after a CL clustering process, $CT(j,:)$ be cluster representative's time vector, $|U_j|$ be the number of its members and h, g denote the 1st and $|U_j|$th member of the U_j cluster respectively. Given that the time objective function E_t is the sum of the normalized distances between users and their cluster representative, it holds that:.

$$0 \leq E_t \leq n \tag{8}$$

Proof: The proof of the E_t scaling is similar to the Lemma 1 proof. ■

Based on the above Lemmas 1 and 2, it holds that both E_p and E_t, defined in (5) and (6) respectively, are of the same scale and thus we can use E, defined in (4), to guide the TUNING TIME-AWARE clustering.

B. The BINDING TIME-AWARE CLUSTERING problem

To capture the two aspects of the time-aware clustering in a binding way, we must define an objective function that incorporates both of them. The motivation is that we can express users' preferences over both pages and timeframes by exploiting the three dimensional structure PTV (n x p x t). The PTV table carries information for page and time users' preferences simultaneously as we extensively presented in Section II.

Let us consider an arbitrary cluster U_j, where $j = 1, \ldots, k$, of the users' set U in the n x p x t space. The representation of the cluster U_j when a clustering process CL is applied to it, collapses the users belonging to U_j into a single three dimensional point (i.e. the mean value which does not correspond to an existing user). We call this point cluster's representative c_j, similarly to the previous subsection approach, and given the page-over-time visiting tables of $u_i \in U_j$, we can respectively define the page-over-time visiting table of c_j in the n x p x t space as follows:

$$CPT(j,:,:) = \frac{\sum_{i=1}^{n} f(u_i, U_j) * PTV(i,:,:)}{|U_j|}$$

Since both $PTV(i,:,:)$, where $i = 1, \ldots n$, and $CPT(j,:,:)$, where $j = 1, \ldots k$, are tables, their dissimilarity is measured by their page-over-time distance $d_{pt}(u_i, c_j)$. Considering all clusters, we define the *binding objective function* E_{pt} to be the sum of distances over both pages and timeframes between each user and the representative of the cluster that the user is assigned to.

$$E_{pt} = \sum_{j=1}^{k} \sum_{u_i \in U_j} d_{pt}(u_i, c_j) \tag{9}$$

At this point, we can define the BINDING TIME-AWARE CLUSTERING problem as follows:

*Problem 2 (*BINDING TIME-AWARE CLUSTERING): Given a set U of n users in n x p x t space, an integer value k, and the binding objective function E_{pt}, find a CL clustering of U into k clusters such that the E_{pt} is minimized.■

IV. TIME-AWARE CLUSTERING ALGORITHMS

Our defined problems are of NP-hard nature since they are a generalization of the well-known clustering problem [20] and thus we can only aim for approximate solutions. Based on the previous section, we define two algorithms to solve the TUNING TIME-AWARE CLUSTERING problem (tuning algorithms) and one algorithm for the BINDING TIME-AWARE CLUSTERING problem (binding algorithm). These algorithms adopt local search heuristics which are similar in spirit with the well-known K-means algorithm [4], which is used for our initial clustering setup. Although K-means does not provide approximation guarantees, it has been proved very effective in many practical problems.

A. Clustering Phases

Our time-aware clustering algorithms are unsupervised, hard partitional methods. The tuning algorithms are used to minimize the objective function E defined in (4) while the binding algorithm aims at minimizing the E_{pt} defined in (9). For each of the three algorithms we adopt a similar, two steps process which is depicted in Fig. 2. During the initialization step which comes after the data preprocessing [21], the algorithms compute an initial set of k clusters guided by the K-means algorithm. The tuning algorithms are based on either page or time users' visiting structures while the binding algorithm initiates clusters using the page visiting structure. Then an iterative reassignment step takes place to improve the initial clusters based on the two requirements: (i) users' page preferences and (ii) the time these preferences were logged.

1) Initialization: We employed the widely used K-means partitional clustering algorithm to produce the initial k clusters. K-means algorithm in summary is: given n points to be clustered, a distance measure d to capture their dissimilarity and the number of clusters k to be created, the algorithm initially selects k random points as clusters' centers and

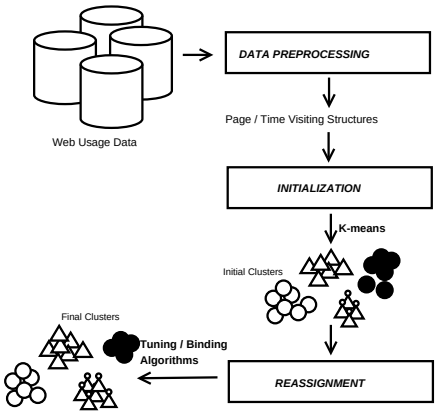


Fig. 2. The time-aware clustering process overview.

assigns the rest of the $n - k$ points to the closest cluster center (according to d). Then, within each of these k clusters the cluster representative (also known as centroid or mean) is computed and the process continues iteratively with these representatives as the new clusters' centers, until convergence.

In our framework, given the n users and the number of k clusters to be created we use CL to denote the initial clustering produced by the K-means algorithm. In case of the tuning algorithms, we adopt two approaches with different logic in the initialization step in terms of prioritizing either page or time aspect. More specifically, the first tuning algorithm initiates clusters based only on the page aspect and thus the CL clustering considers the page visiting structure (i.e. *PV* table) while users' dissimilarity is computed using the d_p distance measure defined in Subsection II-A. Favoring the page preferences would be beneficial, for example, in case of the recommendation engine of a book store, since the topic is more important than the time of the search. On the other hand, in the second tuning algorithm the CL clustering considers the time aspect via time visiting structure (i.e. *TV* table) and uses d_t as users' dissimilarity distance measure (also defined in Subsection II-A). Prioritizing the time aspect is essential, for instance, in building customized pages of a bus info search application interface, since the access time (e.g. bus schedule after 5 p.m. for a user usually searching in the afternoon) is more crucial than the search topic (e.g. bus destination).

The binding algorithm can assign users to the initial clusters taking into account either the users' page preferences or their access time without the clustering process being depended on the aspect considered initially. Since our algorithm aims at advancing earlier approaches, that initiate clusters based only on users' page preferences, it initially considers the page aspect and thus it uses the *PV* table and the d_p distance measure. In both tuning and binding approaches, the initial CL clustering will be the basis for the reassignment step.

2) Reassignment: The reassignment step of all algorithms aims at producing a CL^* clustering which enhances the initial CL to meet the two criteria of the time-aware problems. In the aforementioned examples, it is important to include time in a book store's recommendation engine since users are interested in different books at different time periods (e.g. winter, summer), whereas, in the bus scheduling applications users' destinations (i.e. search topic) is also important for characterizing their profiles. Therefore, given the initial CL clustering, we aim at finding a CL^* that minimizes either the objective function E (4) or the E_{pt} (9) according to the problem to be solved. For example, in the first tuning algorithm which begins considering the page aspect, its reassignment step regards the time aspect. The reverse logic is adopted by the second tuning algorithm. The binding algorithm enhances the initial clustering (i.e. CL based on *PV* table) using the *PTV* table, which incorporates both the page and time aspects, and the d_{pt} distance measure defined in Subsection II-A.

More specifically, the reassignment step begins with the set of k clusters produced by the CL and involves a number of iterations. During each iteration, we compute for each user u_i the fluctuation of the value of the underlying objective function (i.e. E for the tuning algorithms and E_{pt} for the binding algorithm) caused by moving user u_i to one of the rest $k-1$ clusters. If there exist some moves that lead to an improvement in the overall value of the objective function, then u_i is moved to the cluster that leads to the highest improvement. If no such cluster exists, u_i remains in his initial cluster. The reassignment phase follows K-means idea for its convergence, ending either after a number of iterations or when the objective function improvement between two consecutive iterations is less than a minimum amount of improvement specified. In our experiments, we select the number of iterations to be $r = 10$ because we observed that the reassignment step converges in less than 10 repetitions, given that it begins with CL clustering and not randomly as K-means. As a result, we obtain the CL^* with the final clusters.

Our proposed algorithms follow an incremental attitude, since we move a user only when it is determined that such a move will lead to an improvement of the objective function value. Thus, the reassignment phase always converges to a local optimum (i.e. minimum) which depends on the particular cluster representatives selected during the initialization step. To eliminate this sensitivity, we repeat the overall process NUM times and report the best found clustering solution [19]. In our experiments, we select $NUM = 5$ in order to keep low the algorithm's time execution. In addition, according to our observation, in most cases the best local optimum was obtained during the first 5 repetitions.

B. Tuning Algorithms

1) The PAGETUNETIME *algorithm:* The PAGETUNETIME algorithm deals with the TUNING TIME-AWARE CLUSTERING problem giving priority to page preferences and then refining clusters based on time information. The core idea is that we can initiate clustering based only on users' page preferences and then enhance assignment taking into account the time aspect. This algorithm involves the two steps described in Subsection IV-A where, at the first step, an initial CL clustering of the users' set U occurs, based on the n x p page visiting table *PV*. The K-means employed for this CL clustering minimizes the page objective function E_p. Next, given this CL and taking into account the n x t time visiting table *TV*, the clusters' representatives are defined (k x t *CT* table) with respect to

time in order that we compute the time objective function E_t. The calculated E_t is then combined with E_p to form the tuning objective function E. Once the E is initialized, the algorithm proceeds to the second step called the reassignment step. The goal is to find a meaningful users' clustering CL^* so that its tuning objective function E is minimized. The algorithm finalizes the clusters, when all necessary reassignments have been made.

Algorithm 1 The PAGETUNETIME algorithm.

Input: A set U of n users organized in an n x p page visiting table *PV* and an n x t time visiting table *TV* and the number of clusters k.

Ouput: The tuning objective function E and the assignment of the users in the k clusters that minimizes E.

1: */*Initialization Process*/*
2: $(CL, E_p) = K - means(PV, k)$
3: $CT = TimeRepresentative(TV, CL, k)$
4: $D_t = CalcDist(TV, CT)$ */*D_t is an n x k distance table*/*
5: $E_t = CalcObjF(D_t, CL)$
6: $E = \alpha * E_p + (1 - \alpha) * E_t$
7: */*Reassignment*/*
8: $min := E$
9: **for** $r := 1$ to 10 **do**
10: **for** $i := 1$ to n **do**
11: **for** $j := 1$ to $k - 1$ **do**
12: $E^{'} = CheckUserReassignment(i, j)$
13: **if** $E^{'} < min$ **then**
14: $CL^* = PerformReassignment(i, j)$
15: $min := E^{'}$
16: **end if**
17: **end for**
18: **end for**
19: **if** $r > 1$ **then**
20: **if** $EImprovement < 1e - 5$ **then**
21: break
22: **end if**
23: **end if**
24: **end for**

Theorem 1: The PAGETUNETIME algorithm 1 has time complexity $O(n^2)$.

Proof: The K-means algorithm (line 2) used at the initialization phase has time complexity $O(nkm)$, where n is the number of users, k the number of clusters to be created and m the number of iterations that takes the algorithm to converge. However, both k and m are relatively small compared to n and thus, their contribution to the algorithm's complexity can be ignored [13]. So, the initial CL clustering is computed in time linear on the number of users: $O(n)$. *TimeRepresentative* function (line 3) calculates clusters' representatives in $O(k)$ time, since it performs k iterations, one for each cluster. *CalcDist* function (line 4) takes $O(nk)$ time to calculate distances between the n users and k representatives whereas *CalcObjF* needs $O(k)$ time to compute the time objective function E_t of the k clusters. The total time complexity of the initialization process is thus $O(n + k + nk + k)$ which becomes $O(n)$.

During the reassignment process, the outer loop (line 9) is iterated the maximum $O(10)$ times ($r = 10$) while the inner loops (lines 10 and 11) $O(n)$ and $O(k-1)$ times respectively. The *CheckUserReassignment* function (line 12) computes the tuning objective function $E^{'}$ considering that a user moves to another cluster and its time complexity is $O(nk)$ since the calculation of $E^{'}$ is based on the computation of distances between the n users and the k new representatives. If $E^{'}$ is improved (reduced) then the reassignment is actually performed by the *PerformReassignment* function (line 14) which simply assigns the values calculated by *CheckUserReassignment*. The total time complexity of the reassignment phase is thus $O(n(k-1)(nk)) = O(n^2)$.

As a result, the total complexity of Algorithm 1 is $O(n) + O(n^2) = O(n^2)$. ■

2) The TIMETUNEPAGE *algorithm:* The TIMETUNEPAGE algorithm also addresses the TUNING TIME-AWARE CLUSTERING problem but with a reverse logic. The motivation behind this algorithm is that we could start from time locality of users' visits and then enhance assignment taking into account their specific page preferences. This second algorithm also consists of two steps. However, during the initialization step it fixes the initial CL clustering of the users' set U based on the n x t time visiting table *TV*. In this case, the K-means minimizes the time objective function E_t. Once the initial CL clustering of *TV* into k clusters is fixed and E_t is calculated, the algorithm computes the page objective function E_p considering the clusters' representatives (k x p *CP* table) with respect to page visits. The value of E_p is combined with E_t to form the tuning objective function E. Then, the algorithm proceeds to the reassignment step, where the goal is to find a new clustering CL^* of users, that minimizes the value of E. Once the algorithm has decided upon the reassignments that need to be made, it outputs the obtained clusters.

Algorithm 2 The TIMETUNEPAGE algorithm.

Input: A set U of n users organized in an n x p page visiting table *PV* and an n x t time visiting table TV and the number of clusters k.

Ouput: The tuning objective function E and the assignment of the users in the k clusters that minimizes E.

1: */*Initialization Process*/*
2: $(CL, E_t) = K - means(TV, k)$
3: $CP = PageRepresentative(PV, CL, k)$
4: $D_p = CalcDist(PV, CP)$ */*D_p is an n x k distance table*/*
5: $E_p = CalcObjF(D_p, CL)$
6: $E = \alpha * E_p + (1 - \alpha) * E_t$
7: */*Reassignment process of the Algorithm 1*/*

Theorem 2: The TIMETUNEPAGE algorithm 2 has time complexity $O(n^2)$.

Proof: The proof is similar to that of the Algorithm 1. ■

C. The BINDING algorithm

The BINDING algorithm solves the BINDING TIME-AWARE CLUSTERING problem that considers the users' set U in the n

x p x t space and organizes it in the page-over-time visiting table *PTV*. Its first step is similar to that of the PAGETUNETIME algorithm. However, once the initial CL clustering of PV into k clusters is fixed, the algorithm proceeds to the calculation of the binding objective function E_{pt}. More specifically, given the CL clustering and taken into account the n x p x t page-over-time visiting table PTV, it defines the clusters representatives in the n x p x t space (*CPT* table) in order to compute the E_{pt}. The reassignment step of this algorithm aims at finding the CL^* clustering that minimizes the value of E_{pt}. Once the algorithm has converged, it outputs the obtained clusters.

Algorithm 3 The BINDING algorithm.

Input: A set U of n users organized in an n x p page visiting table PV and an n x p x t pave-over-time visiting table *PTV* and the number of clusters k.

Ouput: The binding objective function E_{pt} and the assignment of the users in the k clusters that minimizes E_{pt}.

1: */*Initialization Process*/*
2: $(CL, E_p) = K - means(PV, k)$
3: $CPT = TimeRepresentative(PTV, CL, k)$
4: $D_{pt} = CalcDist(PTV, CPT)$ */*D_{pt} is an n x k distance table*/*
5: $E_{pt} = CalcObjF(D_{pt}, CL)$
6: */*Reassignment process of the Algorithm 1 where E_{pt} is used instead of E*/*

Theorem 3: The BINDING algorithm 3 has time complexity $O(n^2)$.

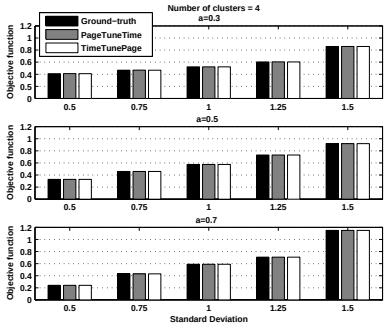
Proof: The proof is similar to that of the Algorithm 1 ■

V. EXPERIMENTAL EVALUATION

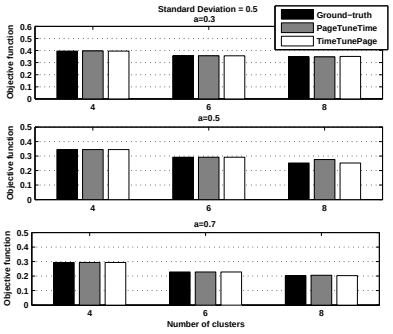
To evaluate the proposed tracks of algorithms we carried out experimentation that involves both synthetic and real datasets. For all cases of synthetic data we found that the proposed algorithms actually "understand" and capture the underlying users behavior model that was initially used to generate the data. For the real data we observed that the proposed time-aware schemes improve the clustering output in terms of the values of objective functions, defined in Subsections III-A and III-B. Furthermore, we studied the impact of time on the clustering output. The results of the algorithms are compared and discussed in order to give an insight of their applicability and importance.

A. Clustering over Synthetic datasets

For the purpose of this experimentation, we have generated data based on a specific model and then tested if the suggested algorithms succeed in discovering that model. More specifically, our synthetic data are generated as follows: initially, we produce the two dimensional page visiting table (the n x p *PV* table) and then based on it we generate the three dimensional page-over-time visiting table (the n x p x t *PTV* table). For the two dimensional table we fix the dimensionality p of the data which was divided in advance into $\frac{k}{2}$ clusters. For each cluster we select a random number of members while for each jth dimension ($1 \leq j \leq p$) we select a mean value $\mu_{i,j}$, which is uniformly distributed in $[0..99]$. Points are then generated by adding a value sampled from the normal distribution $\mathcal{N}(\mu_{ij}, \sigma^2)$. Once the two dimensional page visiting table is created, we proceed to the three dimensional page-over-time visiting table starting by fixing the dimensionality t. The goal here is to split each of the $\frac{k}{2}$ clusters into 2 subsets which will

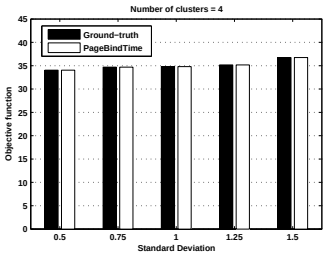


(a) Objective function values as a function of the standard deviation

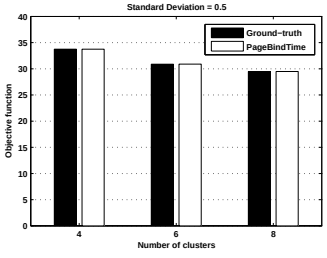


(b) Objective function values as a function of the number of clusters

Fig. 3. Synthetic datasets: PAGETUNETIME and TIMETUNEPAGE algorithms.



(a) Objective function values as a function of the standard deviation



(b) Objective function values as a function of the number of clusters

Fig. 4. Synthetic datasets: BINDING algorithm.

be diversified to the lth dimension ($1 \leq l \leq t$). In this case, we generate values in $[0..t]$ using the normal distribution and use them in order to split each of $PV(i, j)$ value to $PTV(i, j, l)$ values. For our experiments we fixed the values of users' to be around $n = 1000$, $p = 100$ and $t = 10$. We create different datasets using $k = 4, 6, 8$ clusters and standard deviation (for the PV table) $\sigma = 0.5, 0.75, 1, 1.25, 1.5$.

The results for the synthetically generated data are shown in Fig. 3 and 4. These figures depict the objective functions values calculated by the proposed algorithms in comparison with those of the original model, used to generate the synthetic data ("Ground-truth" bar). In all cases, the proposed methods approach the *Ground-truth* objective function demonstrating that our algorithms find the underlying model.

In case of the PAGETUNETIME and TIMETUNEPAGE algorithms (Fig. 3) the *Ground-truth* bar is common since the objective function E of the algorithms is the same. As indicated in Fig. 3(a) the values of objective function are increasing as the standard deviation increases for the different values of factor α $(0.3, 0.5, 0.7)$. The objective function values as a function of the true underlying number of clusters for different α is shown in Fig. 3(b). In this case, it is expected the objective function values be decreasing as the number of clusters increases (this does not always happen since the number of users is random).

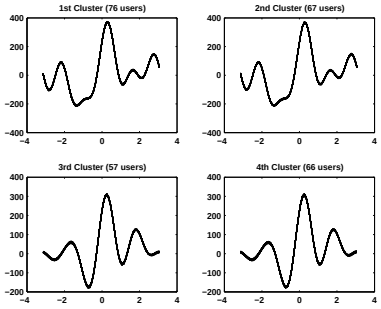
In case of the BINDING algorithm (Fig. 4), the objective function values as a function of the standard deviation are shown in Fig. 4(a) while Fig. 4(b) presents the objective function values as a function of the true underlying number of clusters. The algorithms' performance is apparent in all cases.

1) A Graphical analysis for the synthetic dataset: Graphical analysis is generally very important since it can reveal the underlying structure of a dataset. In case of high-dimensional data, advanced multivariate graphical techniques such as Andrews' curves are employed in order to efficiently depict the data properties [22].

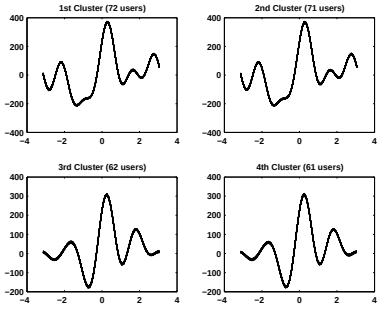
Andrews' curves is a way to visualize and hence to find structure of high-dimensional data. Each multivariate observation e.g. $(PV(i, 1), \cdots, PV(i, p))$ is transformed into a curve based on the function:

$$f(t) = PV(i,1)\frac{1}{\sqrt{2}} + PV(i,2)\sin(t) + PV(i,3)\cos(t) + PV(i,4)\sin(2t) + PV(i,5)\cos(2t) + \cdots$$

and plotted over the range $-\pi < t < \pi$. Thus, each data point (e.g. user) may be viewed as a curve between $-\pi$ and π. This function representation has several interesting characteristics, namely it preserves the standard deviation and the distances of data points (e.g. close points will appear as close curves while distant points as distant curves). So, if there is an underlying structure in the data, it may be visible in its Andrews' curves. More specifically, regarding Andrews' curves in conjunction with clustering process, we can claim that the different shapes of curves among clusters are an indication of dissimilarity between users belonging to different clusters while the similar curves among the users of the same cluster are an indication of similarity between them [23].

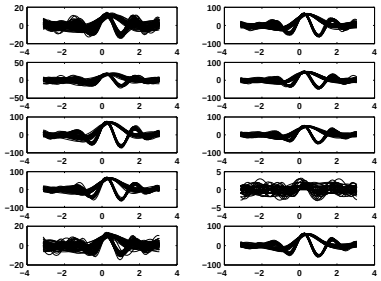


(a) Clusters after initialization phase

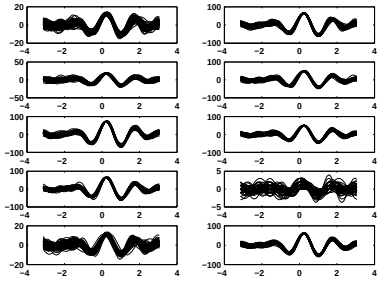


(b) Clusters after reassignment phase

Fig. 5. Andrews' curves in page space: each curve depicts a user based on $p = 10$ pages.



(a) 1st cluster after initialization phase



(b) 1st cluster after reassignment phase

Fig. 6. Andrews' curves in pave-over-time space: for each of $p = 10$ pages each curve depicts a user based on $t = 10$ timeframes.

In our time-aware framework, Andrews' curves can prove graphically the fact that time aspect differentiates users' clusters when it is taken into account in conjunction with their navigational behavior. More specifically, we expect some users

that are considered similar because of their page preferences to be finally grouped separately given the time period of their access. In other words, we expect dissimilar curves within clusters after initialization to be moved in order that we take clusters with similar curves after reassignment. The change in curves will be depicted only when page and time are combined in page-over-time space. So, we study Andrews' curves only with the BINDING algorithm.

In the BINDING algorithm each user can be represented by a single curve based on the p variables: $((PV(i,1),\cdots,PV(i,p))$. In this n x p space, the improvement our approach introduces will not be visible. At the same time, however, we can represent each user with a single curve for each page separately based on the t variables: $((PTV(i,j,1),\cdots,PTV(i,j,t))$, $1 \leq j \leq p$. Here, in n x p x t space, curves shapes after initialization phase are expected to be different within clusters since the aspect of time is not taken into account. The curves' differences are smoothen after the reassignment phase since users' "migrate" among clusters resulting in clusters' members close to each other both in page and time space.

For this part of experiments we set $n = 300$ users, $p = 10$ pages and $t = 10$ timesframes in order to facilitate the readability of the users' graphical representation. We create a dataset using $k = 4$ clusters and standard deviation $\sigma = 1$. As expected, in Fig. 5(a) and 5(b), where each user is represented by a single curve based on $p = 10$ variables, the users' curves have strong similarity within each individual cluster while clusters' curves are different and therefore well-discriminated (we have 2 instead of 4 different curve shapes because we start with 2 actual clusters in page space which are then divided so as to take 4 clusters in page-over-time space). The difference between corresponding clusters after the two phases of the clustering process is on the number of their members. The improvement our approach introduces is apparent in Fig. 6 (due to the lack of space we present only one of the four clusters). Fig. 6(a) and 6(b) represent users in the 1st cluster over each of the $p = 10$ pages. Each user over each page is depicted by a single curve based on the $t = 10$ variables. We can notice that irregularities in initial clustering (Fig. 6(a)) are smoothen after reassignment (Fig. 6(b)). This is a clear indication that our approach creates clusters with members close to each other according to both page preferences and their time locality.

B. Experiments with real data

Our real data experimentation was based on two distinct sources of log files. The first source records users' navigational behavior on an academic oriented host (AUTH CSD Department site[2]) while the second one logs users' visits on a general public, more popular server (NASA[3]). We present experimentation based on two log files derived from the first source and referring to different time periods. More specifically, the first log file involves records over a month period (Oct 03) and its size is about 60MB. The second CSD log file involves records over a seven months period (Oct 03 - Apr 04) and its size is about 430MB. The third log file, derived from the NASA web server, involves records over a month period (Jul 95) and its size is about 200MB. We will refer to these datasets as CSD 1, CSD 7 and NASA, respectively.

As mentioned in Subsection IV-A, the data preprocessing precedes the clustering process and involves data cleaning which removes any log entry that is not needed for the mining process (e.g image files, css, swf or requests made by automated agents and spider programs). Thus, the initial log entries have been significantly reduced so as to work with useful for the clustering information. Table III summarizes the details of the remaining log entries of each dataset. The timeframe in case of the monthly logs (i.e. CSD 1 and NASA datasets) has been set to one day since a day is a typical subdivision of a month's period. On the other hand, in the seven months log (i.e. CSD 7) the timeframe has been defined as one month. However, in the proposed approaches the timeframe's definition is not strict and can be determined according to the underlying application's framework and the log file's period we work on.

TABLE III
DATASETS DETAILS.

Dataset	Time period	Users	Pages	Timeframes
CSD 1	Oct 03	415	113	30
CSD 7	Oct 03 - Apr 04	473	256	7
NASA	Jul 95	456	70	28

In the first part of the real data experimentation, we evaluate the values of tuning E and binding E_{pt} objective functions defined in (4) and (9) respectively. In general, the objective function expresses the sum of distances of each user belonging to a cluster, from the cluster's representative and thus lower values of it declare a better clustering scheme. Consequently, when referring to improvement in terms of objective function we denote its decrement. Table IV presents the improvements' percentages of E and E_{pt} objective functions for the three datasets.

In case of the PAGETUNETIME and TIMETUNEPAGE algorithms, we experimented with different values of the factor α while we set the number of clusters $k = 4, 8, 12, 16, 20$ (for the different values of k the range of improvement percentages is given in Table IV). More specifically, in the PAGETUNETIME algorithm, we notice that the increase of values of α results in lower improvements on clustering objective function E. This is expected since the initial clustering takes into account only the users' page preferences. The tuning objective function E implies that low values of α (i.e. $\alpha < 0.5$) indicate more "gravity" to the time aspect (E_t) and thus result in high levels of improvements while high values of α (i.e. $\alpha > 0.5$) retain "gravity" to the page aspect (E_p) and cause low improvements.

The above observations would be beneficial for applications such as recommendation engines which are primarily based on users' page preferences (that reveal the topics users are interested in) and secondarily should rely on the access time to a different degree. For example, given that users' books

[2]AUTH Department of Informatics, http://www.csd.auth.gr/

[3]NASA server log file, http://ita.ee.lbl.gov/html/contrib/NASA-HTTP.html

TABLE IV
E AND E_{pt} IMPROVEMENTS FOR $k = 4, 8, 12, 16, 20$.

PAGETUNETIME (E)			
Value of α	**CSD 1**	**NASA**	**CSD 7**
0.3	$11\% - 30\%$	$24\% - 45\%$	$28\% - 67\%$
0.5	$4\% - 16\%$	$1\% - 27\%$	$31\% - 55\%$
0.7	$1.5\% - 12\%$	$0.5\% - 20\%$	$18\% - 48\%$
TIMETUNEPAGE (E)			
Value of α	**CSD 1**	**NASA**	**CSD 7**
0.3	0.5%–2.5%	$0.3\% - 10\%$	$6\% - 16\%$
0.5	$4\% - 13\%$	$8\% - 21\%$	$13\% - 43\%$
0.7	$7\% - 25\%$	$15\% - 43\%$	$27\% - 56\%$
BINDING (E_{pt})			
	CSD 1	**NASA**	**CSD 7**
	$28\% - 33\%$	$16\% - 47\%$	$1.5\% - 10\%$

preferences probably vary significantly over time (e.g. job-related in winter, literature in summer), the recommendation engine of a book store should give "gravity" to the time aspect (i.e. $a = 0.3$). This could lead to high levels of improvement, since in our datasets (according to Table IV), the improvements are up to 67%. On the other hand, users' movies preferences are less probable to change over time and thus we can retain the "gravity" on the page aspect (i.e. $a = 0.7$) providing meaningful clusters which in our case are improved up to 48%. In a music recommendation engine, given that users' music preferences are less changeable over time than books' preferences and more changeable than movies' preferences, both aspects would be equally considered (i.e. $a = 0.5$). In this case, our improvements reaching 55% indicate that the PAGETUNETIME clusters are significantly enriched with time information.

The TIMETUNEPAGE clusters also carry more information than those that K-means initially creates as indicated in E improvements (Table IV). In this case, we observe that higher values of α result in higher improvements on E. This is expected since the initial clustering process takes into account only the time locality of users' page preferences. Therefore, choosing appropriate values of α we can either give "gravity" to page information (i.e. $\alpha > 0.5$) or retain "gravity" to time information (i.e. $\alpha < 0.5$).

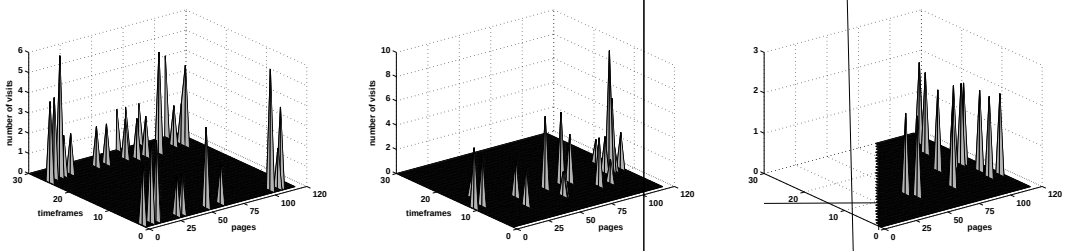
In TIMETUNEPAGE algorithm, similarly to PAGETUNETIME, the degree to which the page aspect should be considered may vary according to the underlying application. For instance, using the TIMETUNEPAGE algorithm with $a = 0.3$ results in improvements up to 16%. This could efficiently guide the administrator of a restaurant's Web site who should update the pages content (e.g. menu) based primarily on users' access time. However, in e-commerce Web sites with marketing campaigns, after deciding the appropriate time for their launching (e.g. at the beginning of a month), it is crucial that the campaign's content be adjusted to users products' preferences. In this case, choosing $a = 0.7$ the TIMETUNEPAGE algorithm could yield improvements up to 56%. Finally, the administrator of a University Department Web server could decide the caching policy according to which pages are mostly visited over specific time periods (e.g. examination results pages cached during the examination period). In this case the page and time aspects would be equally considered (i.e. $a = 0.5$) since this could result in significant improvements which in our case reach 43%.

Comparing the output of the PAGETUNETIME and TIMETUNEPAGE algorithms we can confirm our claim that they are of reverse logic. For example, in case of CSD 1 dataset and for $\alpha = 0.3$ the PAGETUNETIME algorithm results in improvements between 11%-30%, while about the same level of improvements occur in case of TIMETUNEPAGE algorithm when $\alpha = 0.7$ (7%-27%). This observation holds also for NASA and CSD 7 datasets. The variation in the improvement levels between the two algorithms is highly dependent on the initialization. In addition, the different levels of improvements between datasets for the same values of α are due to their nature (i.e. the degree to which the time aspect distinguishes users).

The impact of the time aspect in the reassignment phase of the tuning algorithms is difficult to be represented for all users because of the datasets' high dimensionality. Thus, in Fig. 7 and 8 we focus on the reassignment of a user of the CSD 1 dataset for $k = 8$ in case of the PAGETUNETIME and TIMETUNEPAGE algorithms respectively. More specifically, Fig. 7 depicts the visits of three users as a function of the visited pages and the access time. All three users seem to access common pages. However, the user of Fig. 7(a) requests pages during the first and last timeframes (i.e. days) while the access of the user depicted in Fig. 7(c) is mainly detected on the middle timeframes. Due to their common page preferences, users of Fig. 7(a) and 7(b) are grouped together in U_2 during the initialization phase, even though they differ in terms of their access time. Thus, the reassignment phase of the PAGETUNETIME algorithm manages to separate these users and finally assign user of Fig. 7(b) to cluster U_8 where the user of Fig. 7(c) belongs. It is apparent that these two users are more similar to each other since they pay visits not only to common pages but also on common timeframes. In a similar way, the initialization phase groups together in the cluster U_1 the users of Fig. 8(a) and 8(b) because of their similarity over time. However, due to their dissimilarity on page preferences, the TIMETUNEPAGE algorithm succeeds in reassigning the user of Fig. 8(b) in the cluster U_3 where the user of Fig. 8(c) belongs.

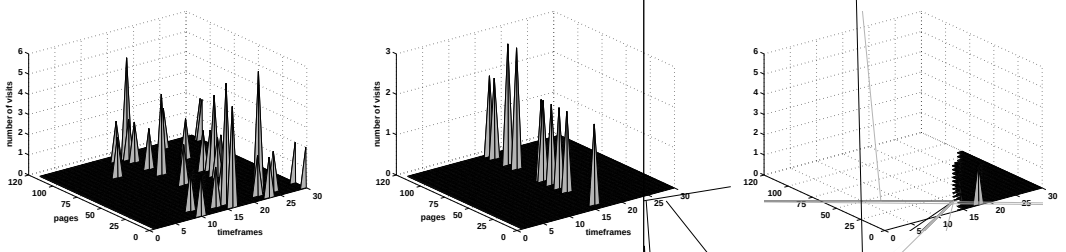
In case of the BINDING algorithm, the improvements intervals on E_{pt} for the various values of k are also depicted in Table IV. The obtained clusters are based equally to the page and time aspect and, as it has been already discussed, this is beneficial for administration issues. For example, the content updating of a Web site as well as caching policies could be guided by users' similar behavior at certain times resulting in refined clusters which in our case are improved up to 47%.

The improvements observed in case of CSD 7 dataset are lower than those of the other two datasets. This fact shows that fine grained timeframes (30 and 28 timeframes in CSD 1 and NASA respectively) provide more information to the BINDING



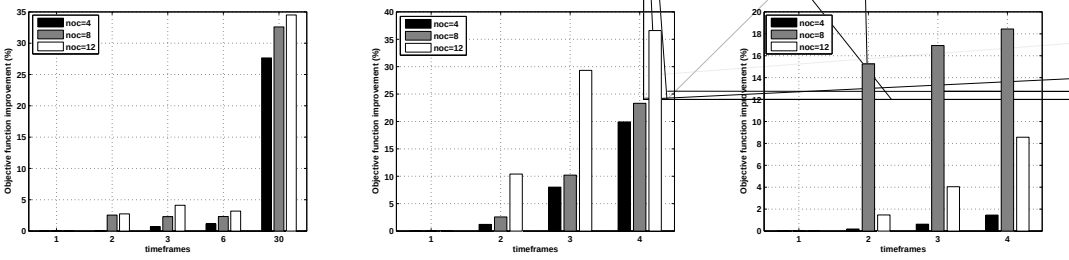
(a) A user of cluster U_2 (b) A user moves from cluster U_2 to U_8 (c) A user of cluster U_8

Fig. 7. The PAGETUNETIME algorithm: the impact of the time aspect in a user's reassignment.



(a) A user of cluster U_1 (b) A user moves from cluster U_1 to U_3 (c) A user of cluster U_3

Fig. 8. The TIMETUNEPAGE algorithm: the impact of the time aspect in a user's reassignment.



(a) CSD 1 (b) NASA (c) CSD 7

Fig. 9. The BINDING algorithm: the E_{pt} improvements as a function of the number of timeframes for $k = 4, 8, 12$.

algorithm and thus they result in higher improvements in comparison with coarse grained timeframes (7 timeframes in CSD 7). This is also demonstrated in Fig. 9 where the improvements of the E_{pt} are presented as a function of the number of timeframes for all datasets for $k = 4, 8, 12$. The number of timeframes on x-axis indicates the time period on whose basis we examine the users' actions. For example, in CSD 1 we define the timeframes $1, 2, 3, 6, 30$ whose duration is $30, 15, 10, 5, 1$ days respectively. It is apparent in all cases that increasing the number of timeframes results in higher improvements. The minimum improvement is 0% when $t = 1$ since in this case the *PTV* table becomes n x p and does not carry any information about time. Fig. 9 indicates that the appropriate timeframe's definition can significantly affect the results of the BINDING algorithm.

C. Interpreting Time-aware Clustering in practice

Analyzing the above derived clusters may yield plenty of inferences about users' preferences and needs which can significantly benefit web-related applications. In accordance to the scenarios highlighted in Section I, the proposed time-aware clustering could be helpful for:

- *E-commerce: targeted market campaigns*, where one can guide certain advertising tasks considering clusters' features [24]. For example, once we use an e-commerce site logs we may have its users clustered according to the TIMETUNEPAGE algorithm. Then a targeted campaign might address different users based on their assignment to clusters and, moreover, non-regular customers might be captured.
- *Recommendation engines*: building users' profiles by analyzing their common browsing behavior can be useful for the effectiveness of recommendation systems which will accurately predict the products (books, movies, music) that a user may be interested in [25]. In such an application, using the PAGETUNETIME algorithm will result in users' clusters that will carry information primarily for users' page preferences and secondarily for their access time. As a result, the recommendation engine could build better correlated and homogeneous profiles.

- *Web site administration*: revealing users' interests can help Web administrators reorganize Web pages' content and layout in order to provide more user-oriented and personalized environment and services [26]. The proposed PAGETUNETIME and BINDING algorithms can be beneficial for such tasks since Web administrators can have clusters containing users with similar preferences and needs at certain times.
- *Caching and Prefetching*: clustering interpretation can provide beneficial indications of Web caching and prefetching since the time criterion is crucial due to space limitations imposed by the cache sizes [27]. Therefore, both the TIMETUNEPAGE and BINDING algorithms can be suitable since they will result in clusters of users who would be coherent in terms of the page and time aspects. Thus, they could offer useful information for effectively prefetching web objects into local caches, reducing latencies and even shifting network loads from peak to non-peak periods.

The proposed algorithm's usage may be extended to any application that can be improved by a users' clustering process, since in most cases the impact of users' visits time locality is crucial and can contribute to revealing more unambiguous relations between them.

VI. CONCLUSIONS-FUTURE WORK

This paper introduces and evaluates two tracks of time-aware clustering approaches, the so-called TUNING and BINDING time-aware clustering. Three different algorithms have been introduced which combine page and time aspects either "loosely" (tuning algorithms) or "tightly" (binding algorithm). The proposed algorithms have been evaluated under real and synthetic workloads and they have resulted in meaningful clusters in terms of the criteria used to evaluate the users' similarity. The produced clusters consist of users exhibiting similar access behavior not only according to their page preferences but also to their access time.

The results of the proposed algorithms offer insight for the adoption of time-aware clustering in various Web applications. Future work may aim at combining the algorithms along with particular caching and prefetching techniques to investigate whether the clustering information could be synchronized with a cache replacement policy. Moreover, these clustering ideas could be used in correlation with particular recommendation engine tasks so as to build better user profiles.

ACKNOWLEDGMENT

The authors thank Dr. Evimaria Terzi for her intuitive ideas and fruitful discussions with respect to the paper's contribution.

REFERENCES

[1] I. Cadez, D. Heckerman, C. Meek, P. Smyth, and S. White, "Model-based clustering and visualization of navigation patterns on a web site," *KDD'03*, vol. 7, no. 4, pp. 399 – 424, 2003.

[2] G. Pallis, L. Angelis, and A. Vakali, "Model-based cluster analysis for web users sessions," in *ISMIS*, Saratoga Springs, USA, May 2005, pp. 219–227.

[3] G. Xu, Y. Zhang, J. Ma, and X. Zhou, "Discovering user access pattern based on probabilistic latent factor model," in *Proc. of the 16th Australasian database conference (ADC '05)*, Darlinghurst, Australia, Jan. 2005, pp. 27–35.

[4] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2001.

[5] Y. Zhao and G. Karypis, "Criterion functions for document clustering: experiments and analysis," Department of Computer Science, University of Minnesota," Technical Report, 2001.

[6] A. Bianco, G. Mardente, M. Mellia, M. Munafo, and L. Muscariello, "Web user session characterization via clustering techniques," in *GLOBECOM'05, IEEE*, Dec. 2005, p. 6.

[7] S. Petridou, V. Koutsonikola, A. Vakali, and G. Papadimitriou, "A divergence-oriented approach for web users clustering," in *Proc. of Int. Conference on Computational Science and its Apllications (ICCSA '06)*, Glaskow, Scotland, May 2006, pp. 1229–1238.

[8] D. Hand, H. Mannila, and P. Smyth, *Principles of Data Mining*. Cambridge, MA, USA: MIT Press, 2001.

[9] M. Albanese, A. Picariello, C. Sansone, and L. Sansone, "Web personalization based on static information and dynamic user behavior," in *Proc. of the 6th annual ACM Int. workshop on WIDM'04*, Washington DC, USA, Nov. 2004, pp. 80–87.

[10] J. Xiao and Y. Zhang, "Clustering of web users using session-based similarity measures," in *2001 Int. Conference on ICCNMC'01*, Beijing, China, Oct. 2001, p. 223.

[11] A. Banerjee and J. Ghosh, "Clickstream clustering using weighted longest common subsequences," in *Proc. of the Workshop on Web Mining, SIAM Conference on Data Mining*, Chicago, 2001, pp. 33–40.

[12] P. Lingras and C. West, "Interval set clustering of web users with rough - means," *Journal of Intelligent Information Systems (JIIS),*, vol. 23, no. 1, pp. 5–16, 2004.

[13] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: A review," *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323, 1999.

[14] D. Knuth, *The Art of Computer Programming*. Reading, Massachusetts: Addison-Wesley, 1973.

[15] R. Duda and P. Hart, *Pattern Classification and Scene Analysis*. New York, NY, USA: John Wiley and Sons, Inc., 1973.

[16] K. Gowda and E. Diday, "Symbolic clustering using a new dissimilarity measure," *IEEE Transactions on Systems Man Cybernetics*, vol. 22, pp. 368–378, 1992.

[17] R. Keeney and H. Raiffa, *Decisions with multiple Objectives: Preferences and Value Tradeoffs*. New York, USA: J. Wiley & Sons, 1976.

[18] K. Schloegel, G. Karypis, and V. Kumar, "A new algorithm for multi-objective graph partitioning," in *Proc. of the 5th Int. Euro-Par'99 Conference Europar*, Toulouse, France, Aug./Sept. 1999, pp. 322–331.

[19] Y. Zhao and G. Karypis, "Topic-driven clustering for document datasets," in *Proc. of the SIAM Int. Conference on Data Mining*, Newport Beach, CA, USA, Apr. 2005, pp. 358–369.

[20] M. Garey and D. Johnson, *Computers and Intractability: A guide to the Theory of NP-Completeness*. New York, NY, USA: Freeman, 1979.

[21] J. Srivastava, R. Cooley, M. Deshpande, and P. Tan, "Web usage mining: Discovery and applications of usage patterns from web data," *SIGKDD Explorations*, vol. 1, no. 2, pp. 12–23, 2000.

[22] D. Andrews, "Plots of high-dimensional data," *Biometrics*, vol. 28, pp. 125–136, 1972.

[23] T. Theodosiou, L. Angelis, A. Vakali, and G. N. Thomopoulos, "Gene functional annotation by statistical analysis of biomedical articles," *Int. Journal of Medical Informatics*, to be published.

[24] S. Boll, "Modular content personalization service architecture for e-commerce applications," in *Proc. of the 4th IEEE Int. Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems (WECWIS'02)*, Newport Beach, CA, USA, June 2002, pp. 213 – 220.

[25] W. Yang, Z. Wang, and M. You, "An improved collaborative filtering method for recommendations' generation," in *Int. Conference on SMC'04, IEEE*, Netherlands, Oct. 2004, pp. 4135 – 4139.

[26] K.-L. Wu, C. Aggarwal, and P. Yu, "Personalization with dynamic profiler," in *Third International Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems (WECWIS'01)*, San Juan, CA, USA, June 2001, pp. 12 – 20.

[27] J. Xu, J. Liu, B. Li, and X. Jia, "Caching and prefetching for web content distribution," *Computing in Science and Engineering*, vol. 6, no. 4, pp. 54 – 59, 2004.

Sophia G. Petridou received the B.S. degree in Computer Science from Aristotle University of Thessaloniki, Greece in 2000, where she is currently working toward the Ph.D. degree in communication networks. Her research interests include clustering, optical and wireless networks.

Vassiliki A. Koutsonikola received the B.S. degree in Computer Science from Aristotle University of Thessaloniki, Greece in 2001, and the M.S. degree in Information Systems from the University of Macedonia, Greece in 2003. Currently, she is a PhD student at Aristotle University of Thessaloniki. Her research interests include clustering, directory services and network-based data organization.

Athena I. Vakali received her Ph.D. in Informatics from the Aristotle University, her M.S. in Computer Science from Purdue University and her B.S. in Mathematics from the Aristotle University. She is currently an associate professor in the Department of Informatics at the Aristotle University, Thessaloniki, where she works as a faculty member since 1997. She is the head of the Operating Systems Web/INternet Data Storage and management research group. Her research activities are on various aspects and topics of the Web information systems, including Web data management (clustering techniques), content delivery on the Web, Web data clustering, Web caching, XML-based authorization models, text mining and multimedia data management. Her publication record is now at more than 100 research publications which have appeared in several journals (e.g CACM, IEEE Internet Computing, WWWJ), book chapters and in scientific conferences (e.g. IDEAS, ADBIS, ISCIS, ISMIS etc). She is a member of the editorial board of the Computers and Electrical Engineering Journal (Elsevier), the International Journal of Grid and High Performance Computing (IGI) and since March 2007, she is the coordinator of the IEEE TCSC technical area of Content Management and Delivery Networks. Prof. Vakali has leaded many research projects in the area of Web data management and Web Information Systems.

Georgios I. Papadimitriou ($M'89$, $SM'02$) received the Diploma and Ph.D. degrees in Computer Engineering from the University of Patras, Greece in 1989 and 1994 respectively. From 1989 to 1994 he was a Teaching Assistant at the Department of Computer Engineering of the University of Patras and a Research Scientist at the Computer Technology Institute, Patras, Greece. From 1994 to 1996 he was a Postdoctorate Research Associate at the Computer Technology Institute. In 1997 he joined the faculty of the Department of Informatics, Aristotle University of Thessaloniki, Greece, where he is currently serving as an Associate Professor. His main research interests include optical networks and wireless networks. Prof. Papadimitriou is Associate Editor of the IEEE Network, the IEEE Communications Magazine, the IEEE Transactions on Systems, Man and Cybernetics-Part C, the IEEE Transactions on Broadcasting, and the IEEE Sensors Journal. He is co-author of the books "Optical Switching" (Wiley, 2007), "Multiwavelength Optical LANs" (Wiley, 2003) and "Wireless Networks" (Wiley, 2003) and co-editor of the book "Applied System Simulation" (Kluwer, 2003). He is author or coauthor of 70 journal and 80 conference papers. He is a Senior Member of the IEEE.