Understanding Computer Usage Evolution

David C. Anastasiu[†1], Al M. Rashid[‡], Andrea Tagarelli[§] and George Karypis[†2]
[†]University of Minnesota, Twin Cities, MN 55455, U.S.A.
[‡]Intel Corporation, 1900 Prairie City Rd., Folsom, CA 95630, U.S.A.
[§]University of Calabria, Rende, Italy
[†1,2]{dragos, karypis}@cs.umn.edu, [‡]al.m.rashid@intel.com, [§]tagarelli@dimes.unical.it

***Abstract*—The proliferation of computing devices in recent years has dramatically changed the way people work, play, communicate, and access information. The personal computer (PC) now has to compete with smartphones, tablets, and other devices for tasks it used to be the default device for. Understanding how PC usage evolves over time can help provide the best overall user experience for current customers, can help determine when they need brand new systems vs. upgraded components, and can inform future product design to better anticipate user needs.**

In this paper, we introduce a method for the analysis of users' computer usage evolution. Our algorithm, `Orion`, segments the application-level usage of different users into a sequence of prototypical usage patterns shared among users, referred to as protos. Following an iterative process, protos are automatically derived from the segmentation, and an optimal segmentation is determined from the protos by a dynamic programming algorithm. To ensure that the segmentation is robust, constraints on the length and the number of segments are utilized.

We show the validity of our method by analyzing a dataset consisting of over 28K users whose PC usage covers approximately 1M weeks. Our results show that different groups of users exhibit different usage patterns, the usage patterns of nearly 50% of the users change over time, and more than 20% of the users undergo multiple changes. Moreover, many of the differences in the usage patterns and their changes appear to correlate with various user-specific information, such as their geographic location and/or the type of computer system that they have. To show the versatility of `Orion`, we present additional results from an analysis of 57K grocery store orders of nearly 1000 users.

I. Introduction

The recent proliferation of computing device form factors leads us to wonder about the role of the personal computer (PC) in our lives today and in the years to come. Understanding how people use computers, how their application-level usage evolves over time, whether and to what extent the usage and its evolution is affected by the form factors and/or the users' location, are all of great interest to designers of PCs and related products. This type of knowledge can be used to provide the best overall user experience for current customers, maintain/improve current product manufacturing, or inform future product design to better anticipate user needs.

In this work, we develop a method to analyze longitudinal multivariate timeseries data pertaining to users' PC usage, with the goal of characterizing and understanding computer usage evolution. The result can be used to identify user populations exhibiting different patterns of evolution, explain how some external factors influence the usage evolution, and suggest the set of capabilities that should be present in future generations of personal computers. Understanding how users' PC usage is evolving, their current utilization and future trends, can help designers create systems better equipped to meet consumer needs, and can provide insights into improving the customer experience for current users.

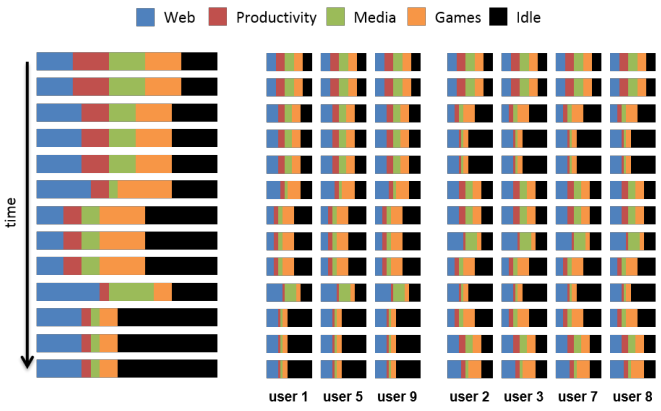

Fig. 1. Computer usage evolution: a user's sequence of PC usage vectors (on the left), and sequences of two similar sets of users (on the right). (Best viewed in color.)

Figure 1 illustrates the idea of computer usage and its evolution by grouping individual applications used by users into high level categories, such as Web, Games, etc. A user's PC usage during a time window, e.g., in a week, may be described by the time she spends on various tasks, including surfing the Web, creating documents or presentations, watching movies or playing games. Each row on the left-hand side of Figure 1 illustrates such a weekly usage pattern represented as vectors. Moreover, a user's behavioral pattern may change over time. For example, our hypothetical user has a decreased overall PC usage as time progresses. Additionally, towards the end, she spends more time surfing the Web, and less time using productivity tools. When we characterize usage evolution of many users, we are able to find groups of users with similar trends (right-hand side of Figure 1). This type of analysis can benefit computer hardware and software designers by providing design feedback and insight into upcoming trends.

Our proposed solution, `Orion`, segments the application-level usage of different users into a sequence of prototypical usage patterns, referred to as *protos*. It then employs an iterative process by which the protos are automatically derived from the segmentation and an optimal segmentation is determined from the protos using a dynamic programming

algorithm. To ensure the segmentation is robust, constraints on the length and number of segments are utilized.

The main contributions of this paper are as follows:

- We describe a method for the analysis of multivariate time series pertaining to resource utilization by users. Our method performs a cross-user usage segmentation, describing user sequences though a small number of protoypical usage patterns (*protos*), which are shared by all users.
- We develop a fully unsupervised, dynamic programming algorithm, named `Orion`, which jointly detects the optimal segmentation and the protos: given the current set of protos, it identifies the segmentation that best encodes user sequences by protos, and, given the segmentation, it identifies the protos that minimize the total error.
- We present results from analyzing a dataset in the PC utilization domain, consisting of over 28K users whose usage covers approximately 1M weeks. Our results show that different groups of users exhibit different usage patterns, the usage patterns of nearly 50% of the users change over time, and more than 20% of the users undergo multiple changes. Moreover, many of the differences in the usage patterns and their changes appear to correlate with various user-specific information such as their geographic location and/or the type of computer system they have.
- `Orion` is versatile and can be applied to diverse multivariate timeseries domains. We demonstrate this through a short analysis of purchase habit evolution of nearly 1000 users at a grocery store.

The remainder of the paper is organized in the following manner. Section II summarizes related work, focusing on existing approaches to multivariate time series segmentation. Section III models the problem and introduces notation used throughout the paper. Section IV details our approach to characterizing behavior evolution. We describe our evaluation methodology and analyze experimental results in Section V. Section VI concludes the paper and also provides pointers for future research.

II. Related Work

The general problem of time series segmentation[1] has attracted a lot of attention from different research communities, including signal processing, pattern recognition, machine learning, and language processing. If no constraints are imposed between different segments, finding the optimal segmentation into m segments can be solved in polynomial time in the order of $O(n^2m)$ by using dynamic programming, where n is the length of the time series. Approximation algorithms with provable error bounds (i.e., theoretical upper bounds for the error compared to the optimal error) have been developed to solve the problem in subquadratic time (e.g., [1]).

Generalizing the time series segmentation to multiple time series variables adds new challenges, mainly related to the definition of segment across the different variables and to scale issues. Like the univariate case, solutions to the multivariate time series (MTS) segmentation problem have been developed for emerging applications in pattern recognition [2]–[5], signal processing [6], [7], and biological systems [8], [9]. Most existing approaches to MTS segmentation fall into three main categories: (i) statistical latent process models, (ii) clustering-based methods, and (iii) dynamic programming.

Statistical latent models for MTS segmentation treat the MTS data as belonging to a particular class of random processes, by which the mutual correlations between MTS data need to be captured while taking into account the temporal constraints within the single time series. Hidden Markov models (HMMs) are widely used in human activity recognition [4], [5], [10]. In particular, Chamroukhi et al. [5] propose the use of a multiple regression model incorporating a hidden discrete logistic process for the recognition of human activity switching over time. A dedicated expectation-maximization algorithm is developed to learn the model, where the number of latent activities is estimated by means of a penalized likelihood criterion. Discrete-time Poisson counting processes are also considered, such as in the astronomy domain. Dobigeon et al. [7] explore joint segmentation of multiple signals coming from different astronomical sensors. Segmentation is performed by applying a hierarchical Bayesian approach to a piece-wise constant Poisson rate model. Markov Chain Monte Carlo methods are used to draw samples according to the posterior distributions. The approach is non-parametric but requires Gibbs sampling to jointly estimate the unknown parameters and the hyperparameters of the model, which may introduce limiting computational time issues.

Clustering-based segmentation approaches employ a customized clustering process which imposes additional constraints, requiring members of a cluster to be contiguous in time and data within segments to be homogeneous. These approaches are hybrid, as they usually combine a clustering algorithm to detect the segment representatives with maximum likelihood estimation [3], [11]–[13]. For instance, the fuzzy maximum likelihood clustering algorithm by Abonyi et al. [3] jointly detects segments based on a probabilistic PCA model and fuzzy sets that represent segments in time. Data are modeled as mixtures of multivariate Gaussian models.

Dynamic programming (DP) has been widely used in segmentation problems [14]–[16]. Recently, there has been a renewed interest in DP approaches to solve MTS segmentation problems. Guo et al. [17] introduce a threshold autoregressive model to define the segment error function of the optimization. However, choosing the ideal value of both segmentation order and autoregressive order is non-trivial. To address this issue, the order of autoregression and segmentation can be simultaneously determined based on Schwarz's Bayesian information criterion. The latter is typically used to estimate the number of segments in multiple change-points problems and, although it may work well in practice, no guarantee is given about the quality of the estimation. Other approaches, such as that by Omranian et al. [9], formulate the MTS segmentation as

[1]Not to be confused with time series summarization or approximation.

a bi-optimization problem: given the time series length n and $O(n^3)$ positive real values for the network-based similarity/distance values computed over all possible segment pairs, find the partition with minimum number of segments which maximizes the sum of distances over all consecutive segments. By transforming the above formulation into a directed acyclic graph (DAG), the problem is solved by determining (via dynamic programming) the maximum weight path with the smallest length in the DAG. The algorithm's performance can be improved by requiring that the segment length is above a given threshold, e.g., adding a breakpoint-penalty for the inclusion of a new segment (breakpoint) to the optimal path.

Our proposed approach falls into the DP category, although it is also coupled with a centroid-based partitional clustering algorithm to produce the proto vectors. As such, `Orion` does not suffer from typical issues of statistical latent models, which generally rely on a model-class assumption of the data and do not automatically provide information on the significance of the estimated parameters. Compared to the DP approaches mentioned above, our work focuses on the definition of a prototypical usage vector (proto) and on the development of an algorithm for optimal proto-based cross-user usage segmentation that were not explored in previous works. Moreover, `Orion` remains quadratic in the (average) time series length, while a network-based approach like [9] has overall complexity of at least $O(n^3)$. `Orion` works in Euclidean space (which supports versatility in practice), while the approach in [9] strongly relies on the local/global centrality measure used to determine the similarity/distance values for all segment pairs. Additionally, the vector-space model we adopt for the representation of the protos does not incur interpretation issues of the estimated parameters of any segment to be detected (e.g., autoregressive coefficients in [17]).

It is worth mentioning that our work is also related to research in dynamic user behavior analysis, particularly for behavioral targeting, which has been a prolific area in Web data mining settings [18]–[20]. The changes in recent users' interests are modeled for improving effectiveness in audience targeting, e.g., for advertising campaigns. However, while also dealing with large amounts of high dimentionality data, and hence sharing our need to refine and concisely model the data into user profiles, online behavioral targeting approaches mostly focus on feature selection/weighting and the predictive capabilities of the profiles. With this purpose in mind, when collecting the users' online behavioral history, fine-grain information on active as well as passive behavioral types are usually available; by contrast, in our setting, predicting the application-usage category is a much harder task due to incomplete information on many executables.

III. Modeling resource utilization

Manufacturers at times collect data related to how their devices or resources are used, with user consent. These data are often multivariate in nature, and observed over a given time span. In the case of PC usage, different users use subsets of applications for arbitrary amounts of time and intensity. Given an analysis time period, we split time into m equally-sized windows, and aggregate (sum up) individual user usage of applications within each window. Let $\boldsymbol{w}_j = [a_1, \ldots, a_q]$ be a usage vector in the application space, i.e., a_k is the amount of CPU time consumed by the k-th application while the user u used that application in the j-th time window.

We follow a segmentation based approach in order to characterize computing usage evolution. Let $\mathbf{T}_u = \langle \boldsymbol{w}_1, \ldots, \boldsymbol{w}_{n_u} \rangle$ be the sequence of PC usage vectors for user u, of length n_u. Both the number of usage vectors and the length of each vector may vary from user to user. Furthermore, $\boldsymbol{w}_j$ and $\boldsymbol{w}_{j+1}$ may not be adjacent in time, only arranged in increasing date order. A segmentation $\mathbf{S}_u$ of length m_u of user u's PC usage sequence is a partitioning of $\mathbf{T}_u$ into m_u non-overlapping contiguous segments that span the entire sequence:

$$S_u = (s_0, s_1, \ldots, s_{m_u}),$$

with $0 = s_0 < s_1 < \ldots < s_{m_u-1} < s_{m_u} = n_u$. The intervals $[s_0+1, s_1], [s_1+1, s_2], \ldots, [s_{m_u-1}+1, s_{m_u}]$ denote the *segments*, and m_u represents the number of segments, or *segmentation order*, for user u's usage segmentation. Note that each user's sequence may be encoded by a different number of segments. Time points $s_0, \ldots, s_{m_u}$ are called segment boundaries, or *change points*, and correspond to changes in behavior of $\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_{n_u}$.

We seek to find a segmentation such that usage in each segment remains fairly consistent. Usage vectors in a segment can then be approximated by a prototypical usage vector (*proto*). Given a small number of protos shared among all users, a *proto-based segmentation* of a user's sequence is one that minimizes the error associated with modeling the segments by the protos. It optimizes a function of the form,

$$\min_{s_*, \boldsymbol{p}_l} \sum_{l=1}^{m_u} \sum_{j=s_{l-1}+1}^{s_l} ||\boldsymbol{w}_j - \boldsymbol{p}_l||^2,$$

where $\boldsymbol{p}_l$ is the proto associated with segment l, and $||\cdot||$ is the vector ℓ^2-norm. The vector $\boldsymbol{p}_l$ captures the consistent usage during $\langle \boldsymbol{w}_{s_{l-1}+1}, \ldots, \boldsymbol{w}_{s_l} \rangle$.

Given segmentations found for all n users, the protos that best approximate original data minimize the total error,

$$\min_{s_*, m_*, \boldsymbol{p}_*} \sum_{i=1}^{n} \sum_{l=1}^{m_i} \sum_{j=s_{i,l-1}+1}^{s_{i,l}} ||\boldsymbol{w}_{i,j} - \boldsymbol{p}_{i,l}||^2.$$

Overall, the unknowns in our model are p proto vectors, and, for each user, their segmentation $\mathbf{S}_u$ and its length m_u.

Table I provides a reference for symbols used throughout the paper. While our discussion focuses on computer usage, note that the method is not restricted to this domain. Vectors $\boldsymbol{w}_i$ can describe other kinds of multivariate observations over time. In the purchase habit evolution experiment we describe in Section V-E, e.g., a_k represents the overall price paid by a customer u for the k-th product in the j-th grocery order.

TABLE I
SYMBOLS USED THROUGHOUT THE PAPER

Symbol	Description
n	number of sequences/users
p	number of protos
q	number of features, e.g., PC applications
$\mathbf{T}_u$	sequence of observation vectors for user u
$\boldsymbol{w}_j$	j-th observation vector in a sequence
$\boldsymbol{w}_{j,i}$	j-th observation vector in i-th user sequence
a_k	element of observation vector, e.g., application CPU time
n_u	number of observation vectors for user u
$\mathbf{S}_u$	segmentation of $\mathbf{T}_u$
s_l	l-th segment in a segmentation
$s_{i,l}$	l-th segment in i-th user segmentation
m_u	number of segments in $\mathbf{S}_u$
$\mathbf{P}$	current proto vectors
$\boldsymbol{p}_l$	proto encoding l-th segment
$\boldsymbol{p}_{i,l}$	proto encoding l-th segment in i-th user segmentation
α	minimum segment length during segmentation
β	cost for creating a new segment

IV. ORION

We model each user's computer usage as a sequence of prototypical usage vectors (protos). Given a small number of protos, `Orion` performs a cross-user usage segmentation, i.e., a segmentation of the sequences of all users such that the error associated with modeling each segment by one of the protos is minimized. We find the cross-user usage segmentation in an iterative manner. In the *segmentation* phase, we compute an optimal proto-based encoding of the user's multivariate sequence via a dynamic programming algorithm. In turn, in the *update* phase, protos are informed by the usage vectors in the segments they model within all user sequences. The process iterates until the reduction in error is small enough or a user-defined number of iterations has been reached.

A. Segmentation

A segment is well approximated by a proto when the sum of the (squared) Euclidean distance between the proto and each of the usage vectors within the segment is small. Initial protos are determined as the centroids of a K-means clustering of all usage vectors across all users. Then, at each iteration, given the current set of protos, we use a dynamic programming algorithm to identify the optimal segmentation minimizing the sum of those segment errors. To ensure robustness of the result, we enforce a minimum length constraint α on each segment and assign a penalty β associated with the creation of each additional segment within a user's sequence. In other words, a segment is allowed to be created if it meets a minimum length constraint and leads to a user-specified reduction in the approximation error. For the remainder of our discussion on segmentation, we will drop the user subscript u to simplify notation.

Following notation by Kehagias et al. [21], we define the *segmentation cost*,

$$J(S) = \sum_{k=1}^{m} d_{s_{k-1}+1,s_k},$$

where m is the number of segments and $d_{e,f}$ (for $0 \leq e \leq f \leq m$) is the segment error corresponding to the segment $[e, f]$. Obviously, the segment error $d_{e,f}$ depends on the data $\mathbf{T} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_n]$ and the model chosen to approximate the data. Given our goal to characterize the usage sequence by a sequence of protos, we define the *segment error* as the reconstruction error,

$$d_{e,f} = ||\mathbf{T}_{e,f} - \hat{\mathbf{T}}_{e,f}||_F^2,$$

where $\mathbf{T}_{e,f}$ is the matrix formed from columns $e, e+1, \ldots, f$ of $\mathbf{T}$, and $||\cdot||_F$ is the matrix Frobenius norm. Here, $\hat{\mathbf{T}}_{e,f}$ is the best usage approximation of the segment that could be obtained via one of a small p number of prototypical usage vectors,

$$d_{e,f} = \min_l d_{e,f}(l) = \min_l \sum_{j=e}^{f} ||\boldsymbol{w}_j - \boldsymbol{p}_l||_2^2.$$

We denote by $d_{e,f}(p)$ the reconstruction error for segment $[e, f]$ obtained using the proto with ID p.

The additive segment cost formulation allows solving the segmentation problem optimally and efficiently via dynamic programming [17], [21]. As we increase the number of segments m, however, even beyond the true number of segments, the segmentation error decreases. To ensure the segmentation is robust, we introduce a minimum segment length constant α and a segment creation penalty β in the segmentation procedure. The optimum segmentation is then described by the recurrence relation,

$$C_f = \min\Big(d_{1,f} + \beta, \min_{\alpha<j<f-\alpha}\big[C_j + d_{j+1,f} + (L_j+1)\beta\big]\Big),$$

where C_x is the optimum segmentation cost of the segment $[1, x]$, and L_x is the number of segments for that optimum segmentation. The first term in the outer $\min$ represents the cost of representing the entire $[1, f]$ sequence as one segment. The last term in the expression on the second line, $(L_j + 1)\beta$, represents the penalty for creating an additional segment. The formula implies that $f - e \geq \alpha$, for all consecutive segmentation points $e, f \in S$.

Algorithm 1 describes our proto-based dynamic programming segmentation of sequence $\mathbf{T}$ of length n with minimum segment length constraint α and segment creation cost β. Segments in the resulting segmentation are approximated by one of p protos, which are the columns of $\mathbf{P}$. We use c_x to capture the segmentation error of the optimum segmentation of segment $[1, x]$, which has order m_x. In this segmentation, p_x identifies the proto that was used to approximate its last segment and z_x shows the index where that segment starts. We say that proto p_x *encodes* the segment. Due to the minimum length constraint, in the initialization phase of the algorithm, we encode segments of length less than 2α by their best approximating proto. In the minimization stage, we find the optimum segmentation for a segment $[e + 1, f]$ of length at

Algorithm 1 Dynamic programming segmentation with minimum segment length α and segment creation cost β.

1: **function** SEGMENTSEQUENCE($\mathbf{T}, \mathbf{P}, n, \alpha, \beta$)
Initialization:
2: **for each** $f = 1, 2, \ldots, n$ **do**
3: $c_f = \infty;\ p_f = -1;\ n_f = 1;\ z_f = 0$
4: **end for**
5: **for each** $f = \alpha, \alpha + 1, \ldots, 2\alpha - 1$ **do**
6: $\epsilon_f = \min_p d_{1,f}(p)$
7: **if** $\epsilon_f + \beta < c_f$ **then**
8: $p_f = \text{argmin}_p\, d_{1,f}(p)$
9: $c_f = \epsilon_f$
10: **end if**
11: **end for**
Minimization:
12: **for each** $f = 2\alpha, 2\alpha + 1, \ldots, n$ **do**
13: **for each** $k = 1, 2, \ldots, p$ **do**
14: **for each** $e = f - \alpha, f - \alpha - 1, \ldots, \alpha + 1$ **or** 0 **do**
15: $\epsilon_f = c_e + d_{e+1,f}(k)$
16: **if** $\epsilon_f + (n_e + 1)\beta < c_f$ **then**
17: $p_f = k;\ c_f = \epsilon_f$
18: $z_f = e;\ n_f = n_e + 1$
19: **end if**
20: **end for**
21: **end for**
22: **end for**
Backtracking:
23: $S(0) = 0$
24: **for** $k = n,\ k > 0$ **do**
25: $S(n_k) = k$
26: $P(n_k) = p_k$
27: $k = z_k - 1$
28: **end for**
29: **return** $S,\ P$

least α based on the already found optimum segmentation of the segment $[1, e]$. Note that the sequence may be best encoded by a single proto when the minimum cost c_f is encountered for $e = 0$ in line 14. In the backtracking stage, we collect and then return the segmentation points S and the associated list of encoding protos P of the optimum segmentation.

B. Update

In the *segmentation* phase, we found, for each user, an optimal proto-based encoding of their multivariate computer usage sequence. Given these segmentations, in the *update* phase, `Orion` identifies protos that minimize the total error,

$$\min_{s_*, m_*, p_*} \sum_{i=1}^{n} \sum_{l=1}^{m_i} \sum_{j=s_{i,l-1}+1}^{s_{i,l}} ||\boldsymbol{w}_{i,j} - \boldsymbol{p}_{i,l}||^2.$$

These vectors are the mean of the usage vectors spanned by the protos.

C. Analysis

The model described in `Orion` makes several assumptions. First of all, we assume that different users exhibit a rather small number of prototypical usage behaviors, which are captured by the protos. Secondly, we assume that the usage behavior of users remains consistent over a certain period of time. Finally, we assume that the usage behavior of users can change from one prototypical behavior to another. As we have discovered, which we will further show in section V, these assumptions are not far-fetched. In many cases, user sequences were able to be described by 2-4 protos, as shown in sparse form in the example below.

User 1: $\langle p_1 : 15, p_5 : 11 \rangle$
User 2: $\langle p_2 : 5, p_3 : 10, p_2 : 7, p_5 : 22 \rangle$
User 3: $\langle p_1 : 11, p_4 : 15, p_5 : 40 \rangle$
User 4: $\langle p_1 : 13, p_5 : 25 \rangle$

`Orion`'s runtime is dominated by the segmentation step, which leads to a complexity of $O(n \times p \times \mu^2)$, where μ is the average user sequence length, $\mu = (1/n) \sum_u n_u$. In practice, we have observed the algorithm converges quickly and only a small number of *segmentation-update* iterations are necessary, typically less than 20.

V. Experimental Evaluation

`Orion` was designed to describe user behavior evolution, which we evaluate in two domains. We focus most of the discussion on user PC usage, and then further demonstrate `Orion`'s utility through an analysis of customer purchase behavior changes at a grocery store.

A. Data Processing

The PC usage data we analyze in this paper are generated from an anonymous data collection project run jointly by Intel and its PC OEM partners. The project aims to understand user experience and issues (including performance), user needs, and how users use their computers, with a goal of improving product design. Anonymous behavioral data is collected from the user systems where the owners explicitly opt in. No personally identifiable information, such as email addresses, names, system serial numbers, or MAC addresses are collected. As of this writing, about 15M systems worldwide have been sending structured data, amounting to about 30TB in relational databases. Captured information includes system type, geolocation (at the country level), CPU type, temperature, battery, on-off behavior, application usage, etc. However, these usage data cannot be tied back to a particular user, unless, for example, the user provides some generated identifier during a customer service call. We will refer to the dataset derived from this project as the PC behavioral dataset (PCB).

Each user's data in the PCB dataset is in the form of a daily application usage summary. For each application it describes, the summary shows execution start and end time, CPU time (broken as user and system process times), number of page faults, etc. We focused our analysis on application CPU time, which we computed as the sum of system and user level process times. Data is not necessarily received each day for each user, due to a number of factors such as network transmission errors, the user's computer being turned off, or the client-side data collector being offline. As a result, user sequences are sparse and may be missing data for days or

weeks at a time. We focus our analysis on *weekly* aggregated application CPU time usage information for each user.

The PCB dataset contains usage data for a random subset of 250K users, spanning 100 weeks, and resulting in an initial 7.52B utilization records. A utilization record shows the CPU time used by application a on user u's computer within one day. We aggregate data at the week level, using calendar weeks as a reference. Figure 2 shows the reduction in the number of records after the weekly application utilization aggregation. The deep valeys show weeks when data was received for fewer users.

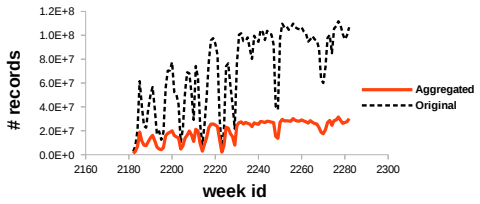

Fig. 2. Weekly aggregation of utilization records.

For the purpose of our analysis, the actual week when data was received for a user is unimportant, as we focus on behavioral changes in the user's utilization sequence. We therefore concatenate each user's sequence of observations, i.e., remove weeks with no utilization from their sequence. Most users, however, have records for at least 50 weeks of PC usage, as shown in Figure 3 (left).

Data collected is very noisy. For example, 1.49B of the initial 7.52B records contained no utilization information, i.e., executables did not consume any CPU cycles in these cases. Figure 3 (right) shows the number of "empty" records per week. We further filtered records for applications that could hinder our analysis. The "idle" process, in particular, records as "CPU time" the number of seconds that the CPU is *not* used. An initial classification of application names was perfomed, assigning applications into one of 10 base classes, including "Office", "Internet", "Util", "Game", "Communication", "Anti-Virus", "System/Other", etc. Roughly 10% of the records belong to applications that could not be properly classified. We removed them, along with "Anti-Virus", "System/Other", and "Util" category records, as they do not provide information about applications directly used by users. Additionally, we removed records for the "Internet" application category, as we found it did not provide salient usage behavior information. The use of a Web browser is currently a "black-box", and does not tell us whether the user is watching shows, playing games, being productive, or reading a book online.

To ensure enough information was present for important enough applications to be modeled by `Orion`, we further removed records with less than 60 seconds of utilization in a week, and removed applications with less than 2K records over the entire user population. We then focused our analysis on users with at least 5 records per week in at least 20 weeks. Table II shows the final user, application, and weekly

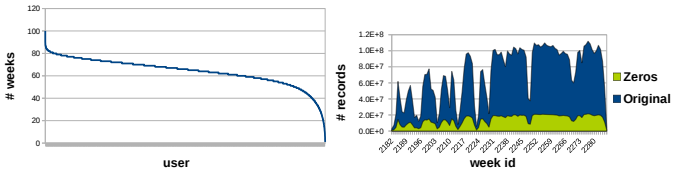

Fig. 3. Distribution of number of weeks with data for users (left). Number of records with no utilization information (right).

TABLE II
STATISTICS FOR PCB DATASET ANALYZED BY ORION

category	count
users	28360
applications	762
weeks	100
records	11.05M

application usage record counts, after filtering, for the PCB dataset.

B. Proto description

We used `Orion` to determine prototypical usage behaviors for more than 28K users over a span of 100 weeks. In our experiments, we used $p = 15$ protos, $\alpha = 5$ minimum segment length, and assigned a penalty $\beta = 0.01$ for creating new segments. In order to dampen the large range of application usage times among different applications and users, we took the log of all utilization values a_k in our PC usage analysis.

The protos identified by `Orion` are quite informative. We present them in Figure 4, further sub-divided into four categories: productivity, communication and media, Asian applications, and gaming. Remember that a proto is, in essence, the centroid of the set of weekly application utilization vectors encoded by the proto after segmentation across all users, and is thus not normalized. The intensity score $int(\boldsymbol{p})$ is the squared length, or ℓ^2-norm, of proto $\boldsymbol{p}$,

$$int(\boldsymbol{p}) = \sum_{k=1}^{q} \bar{a}_k^2,$$

and measures the magnitude of the application utilization times represented by the proto. Here, $\bar{a}_k$ is the average application utilization for the k-th application across all time slices encoded by proto $\boldsymbol{p}$. We have highlighted some telling intensity scores in red. Along with the proto intensity score, we also show a list of high average utilization applications in each proto. We rank applications based on the percentage of their usage within the ℓ^2 norm of the proto, and display the top ranking results along with thier usage percentage. We use this list of applications to choose an appropriate name for the proto. In the following, we will provide brief descriptions of the protos identified by `Orion` in our analysis.

Work/productivity behaviors.

- $P2$. ***Media creation.*** *photoshop* (Adobe Photoshop) dominates this proto, and is joined by other media creation applications

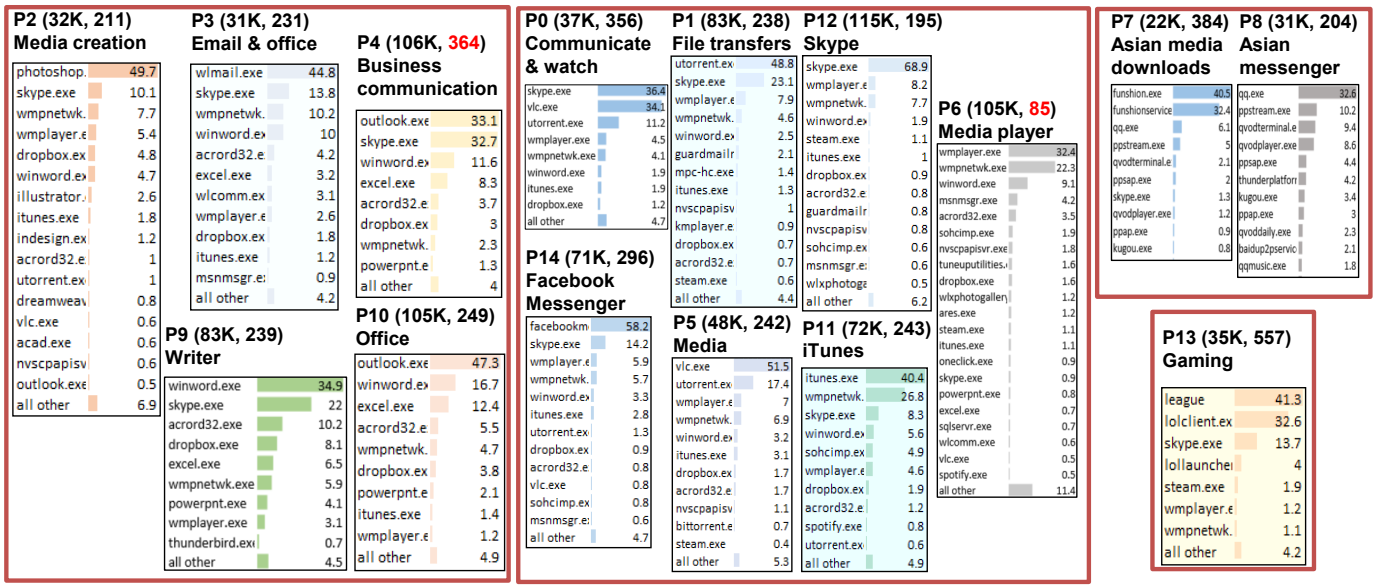

Fig. 4. Protos discovered by `Orion`: work/productivity protos (left), media & social protos (middle), gaming and Asian media & social protos (right). For each proto, we display its identifier, the number of time slices (weeks) the proto has encoded across all users, its intensity score, a given name, and important features. For example, "P2 (32K, 211) Media creation" refers to proto ID P2, which encoded at least 32K weeks and has an intensity score of 211.

such as *illustrator* (Adobe Illustrator), *indesign* (Adobe InDesign), *dreamweaver* (Adobe Dreamweaver), and *acad* (Autodesk AutoCAD).

- $P3$. ***Email & Office.*** *wlmail* (Windows Live Mail) is joined by Microsoft Office applications (*winword*, *excel*) and other productivity software, *dropbox* (Dropbox), and communication software, *skype* (Skype) and *wlcomm* (Windows Live Communications Platform), to describe someone hard at work.
- $P4$. ***Business communication.*** Electronic communication is at the center of activity in this proto, with *outlook* (Microsoft Outlook) and *skype* at the forefront. Electronic office applications, winword, excel, acrord32 (Acrobat Reader), dropbox, and powerpoint (Microsoft Powerpoint), round up the proto. $P4$ achieved the highest intensity score among productivity protos, showing that business communication remains an important function for PCs.
- $P9$. ***Writer.*** *winword* (Microsoft Word) is the predominant application in this proto, joined by several other productivity applications. *wmplayer* (Windows Media Player) has a fairly high score, showing some people likely listen to music while composing documents.
- $P10$. ***Office.*** While $P3$ is more indicative of a home office setup, this proto describes a business office, communicating via *outlook*, using Office applications heavily (*winword*, *excel*, *powerpoint*), along with other productivity software.

Media & social behaviors.

- $P0$. ***Communicate & watch.*** Utilization in proto $P0$ is dominated by *skype* and *vlc* (VideoLAN VLC), a popular media player. Other media applications such as *wmplayer* and *itunes* (Apple iTunes) also play a significant role. While users with this behavior pattern also tend to use peer-to-peer sharing programs like *utorrent*, productivity apps like *winword* are seldom used.
- $P1$. ***File transfers.*** Almost half the activity in $P1$ is peer-to-peer sharing through *utorrent* (uTorrent), with a mix of media playing applications coming second in usage intensity.
- $P5$. ***Media downloads.*** Media playback (*vlc*, *wmplayer*, *itunes*) is the focus of this proto, though downloads, many probably media related, also play an important role through *utorrent* and *bittorrent* (BitTorrent).
- $P6$. ***Media player.*** While *wmplayer* and related services account for half the magnitude in this proto, the remaining applications are less task-focused. The low intensity score achieved by this proto is a further indication that users do not use their PCs heavily while in this state.
- $P11$. ***iTunes.*** The popular media player *itunes* (iTunes) is the focus of this proto. High magnitude for *wmpnetwk* (Windows Media Player Network Sharing Service), combined with fairly low magnitude for *wmplayer*, seems to indicate users in this state stream Windows media content through iTunes rather than Media Player.
- $P12$. ***Skype.*** In this state, communication program *skype* is key, overshadowing all other PC usage.
- $P14$. ***Facebook Messenger.*** With both a high score and a fairly large week frequency count, this proto is indicative of users who communicate most often via Facebook Messenger.

Asian media & social behaviors.

- $P7$. ***Asian media downloads.*** *funshion* (Funshion), a peer-to-peer streaming video player and downloader for East Asia users, and its associated service, dominate this proto. Additional usage is consumed by *qvodplayer* (Nora QvodPlayer), a video-on-demand and overall media player, *ppstream* (PPStream), a Chinese peer-to-peer streaming video network software targetting television content, and their associated services.
- $P8$. ***Asian messenger.*** *qq* (Trencent QQ) is an instant messaging software service popular among Asian users. Asian peer-to-peer media streaming services round up prominent programs in this proto.

Gaming.

- $P13$. ***Gaming.*** *league* (League of Legends), a multiplayer online game, and its clients (*lolclient, lollauncher*), head up utilization, joined by the gaming platform *steam* (GameSpy Steam). While it does not show up in as many weeks as other protos, $P13$ has the highest intensity score of all protos, which points to high PC utilization for its applications.

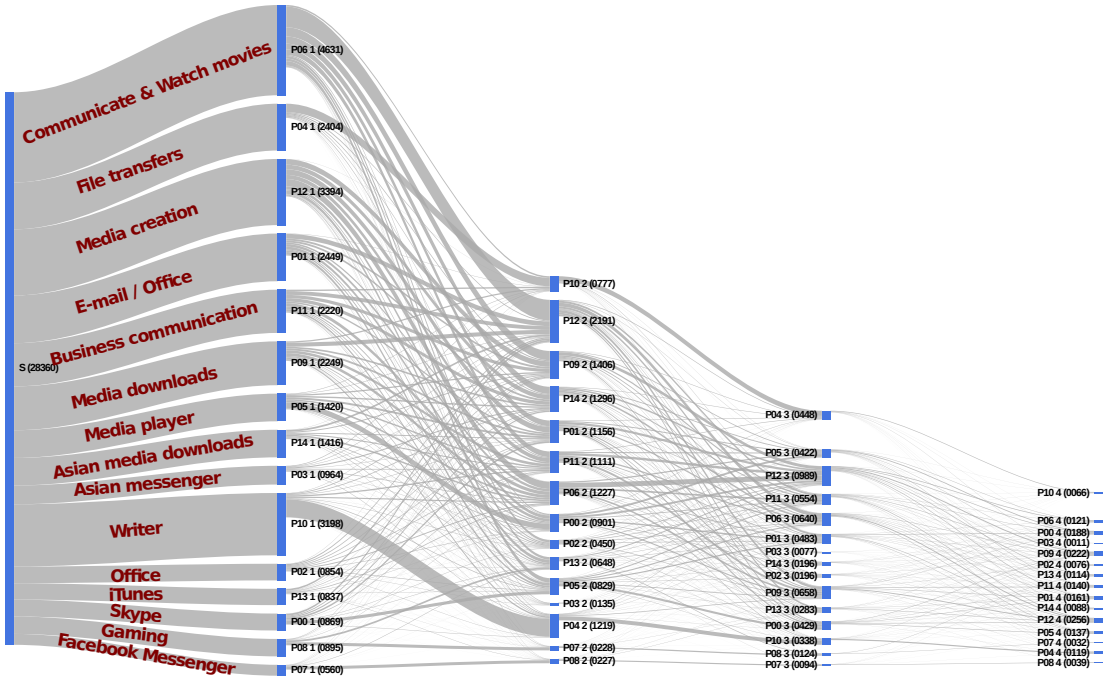

Fig. 5. PC usage proto evolution. Each blue vertical line represents a behavior state, either *Start*, denoted by S, or one of the p protos. Gray lines show users transitioning from state to state. The portion of each state (blue line) that does not have outgoing gray lines are users that end in this state, i.e., transition from here to *End*. Proto states are labeled with their ID, level in sequence, and their frequency, i.e., the number of users that have that proto at that level in their proto sequences. Blue lines are proportional to the proto level frequencies. For convenience, we also show the names of the level-1 protos.

C. Proto evolution

`Orion` transforms PC usage sequences into proto sequences. We can follow along with users as their behavior changes from one state to another simply by following proto-to-proto evolutions in the users' sequences. Figure 5 shows a diagram of the first four levels of proto evolutions in our experiment, for all users. As can be seen from the first two levels of proto evolution, the usage patterns of nearly 50% of the users change over time, and more than 20% of the users undergo multiple behavior changes. Some protos are more stable than others, in the sense that fewer users transition out of that proto state once they enter it. As an example, business users are well represented by the *Office* proto ($P10$), which retains nearly $2/3$ of its users, and transitions the majority of the remaining users to *Business communication* ($P4$). Level-2 *Business communication* has even higher stability, and transitions the majority of its remaining users back to the *Office* state, which again transitions overflow to *Business communication*. A similar back-and-forth transition between two proto states can be seen with *Asian media downloads* ($P7$) and *Asian messenger* ($P8$), indicating many Asian users use both Asian social media and video-on-demand services, alternating intensity of one vs. the other.

Media player ($P6$) and *Skype* ($P12$) are less stable level-1 states and transition large numbers of users to almost all other protos. This seems to indicate they are "interior" points, transition states used by users in-between focussing on other tasks. *Skype* has the highest user count in level-2 and receives large numbers of users from most other protos, indicating online communication is an important user activity alongside other tasks users accomplish using their PCs.

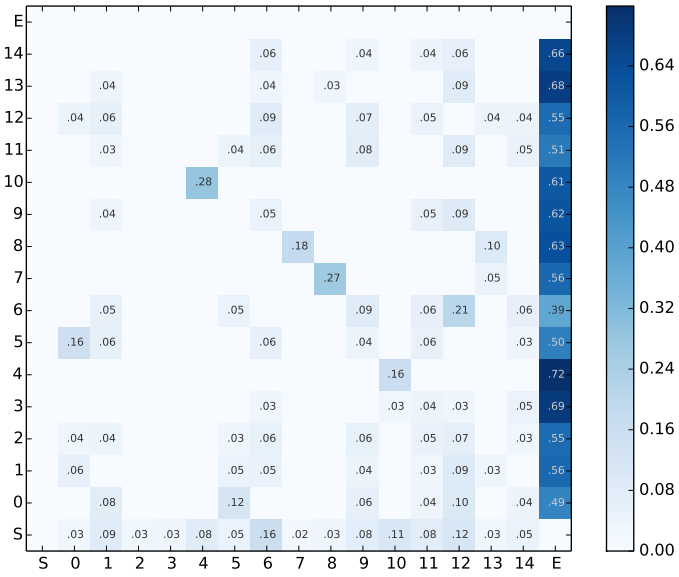

Fig. 6. PC usage proto transitions. S and E are the *Start* and *End* states, and the numbers denote protos. Each row shows the probabilities of a user transitioning from the proto identified by the row ID towards other protos, identified by column IDs, irrespective of level within the user sequences. We only show probabilities above 0.01 in the transition matrix.

While Figure 5 gives us a level-wise view of proto evolution, the proto-to-proto transition probability matrix in Figure 6 shows us the overall picture. We will note by $\Pr(P10, P4) =$

0.28 the probability of a user transitioning from *Office* to *Business communication*, which is located in row ID 10 and column ID 4 in the matrix. Transition probabilities confirm some of the observations we made by analyzing proto evolutions. $\Pr(P6, E)$ and $\Pr(P0, E)$ are both low in comparison to other probabilities of transitioning to the *End* state (column E in the matrix), confirming the status of the *Media player* and *Communicate & watch* protos as "interior" points. Rows 10 (*Office*) and 4 (*Business communication*) only transition to each other and to the *End* state, both having high *End* state transition probabilities, which is indicative of their stability. Similarly, $\Pr(P7, P8)$ and $\Pr(P8, P7)$ are both high, relative to other proto-to-proto transitions, indicating an affinity of a user segment for the *Asian media downloads* and *Asian messenger* prototypical behaviors.

Communicate & watch and *Media downloads* also indicate a strong affinity, given $\Pr(P0, P5) = 0.12$ and $\Pr(P5, P0) = 0.16$ are the highest probabilities in their respective rows other than that of transitioning to the *End* state. This is indicative of PC usage as an entertainment and communication center by some users. *Media player* ($P6$) and *Skype* ($P12$) transitions confirm the finding, $\Pr(P6, P12) = 0.21$ and $\Pr(P12, P6) = 0.09$, though they indicate higher likelihood of PC usage as an electronic communication device.

Media creation ($P2$) and *Email & Office* ($P3$) have very low counts of states transitioning into them (*fan-in*, values along the proto column) and high counts of protos they transition into (*fan-out*, values along the proto row). This seems to indicate they are "early" states the users quickly transition out of towards other behaviors. On the other hand, states with low fan-in and low fan-out, like *Business communication*, *Office*, *Asian media downloads*, and *Asian messenger*, are indicative of nitche user groups, such as business and Asian users.

D. Side information correlation

In addition to application usage daily statistics, the client-side collectors capture, for each user, some general information about the user's system and location, including system type, CPU type, and geolocation (at the country level or above). We can use this side information to find correlations between users and prototypical behaviors. In particular, for each proto transition, we compute the Kullback-Leibler divergence (D_{KL}) between the side information distribution (geolocation, system type, or CPU type) of the users that belong to the "from" proto and the users that transition to the "to" proto.

As an example, let us consider the transition *Start* $\Rightarrow$ *Media creation*, and analyze its *system type* correlation. There are 28,360 users in the "from" state (S), and 854 users in the "to" proto ($P2$). We compute the *system type* probability distributions, Q and P, for these two sets of users, and then compute the KL-divergence as,

$$D_{KL}(P||Q) = \sum_i \ln\left(\frac{P(i)}{Q(i)}\right) P(i).$$

Next, we report results for some proto transitions with relatively high KL-divergence. In each result, we show the "from" and "to" probability distributions, and highlight increased or decreased probabilities for certain side information categories. We denote the *Start* state as *All* in the graphs, since all users participate in this state. Figure 7 showcases the two distributions in our running example side-by-side. The result conforms with one's expectations. Users engaging in the *Media creation* PC usage behavior are more likely to be using a Multimedia PC and less likely to be using an Everyday system than other users.

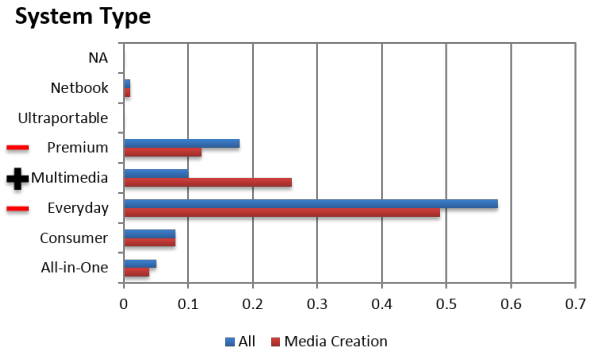

Fig. 7. *Media creation* correlation with *system type*.

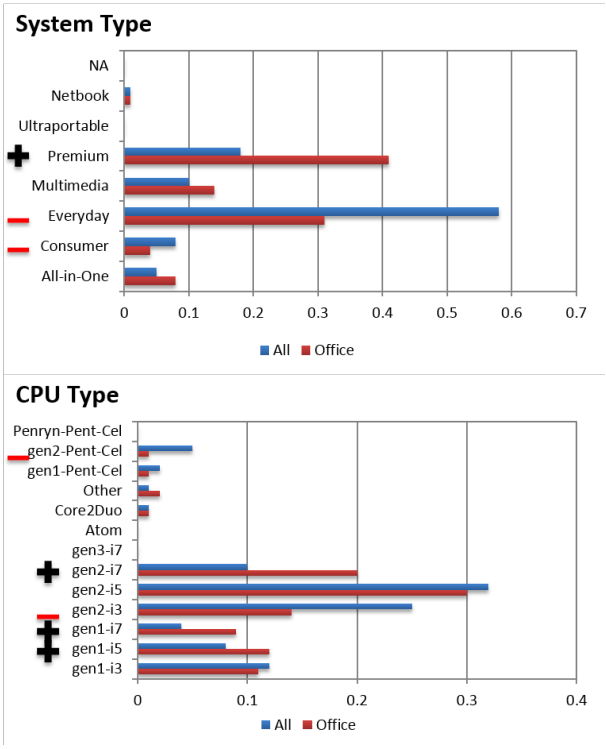

Fig. 8. *Office* correlation with *system type* and *CPU type*.

Figure 8 shows the *system type* and *CPU type* correlation for the *Office* proto state. We show users in this state are more likely to have higher end systems (top figure; Premium, rather than Everyday) with high–end processors (bottom figure; i7 and i5, rather than i3, Pentium, or Celeron). This may

indicate business users may often be using "company owned" machines, which are likely to be more powerful than "home" systems.

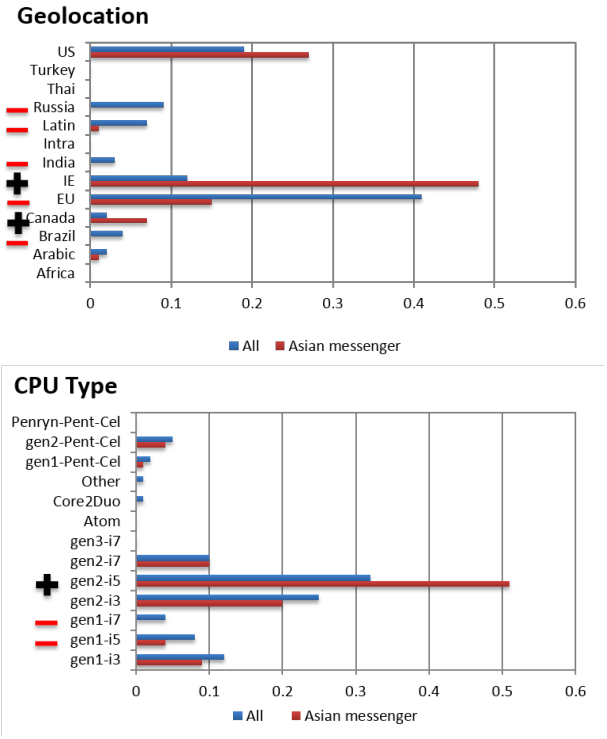

Fig. 9. *Asian messenger* correlation with *geolocation* and *CPU type*.

It should be of no surprise that users in Russia, Latin America, India, European Union (EU), and Brazil are not likely to use *Asian messenger*, likely due to the language barrier. Figure 9 (top) shows the *geolocation* correlation information for *Asian messenger*. United States (US) and Canada, with fairly large Asian minorities, are more likely to have users transitioning into this proto state. IE, which stands for Internation English speakers, compose the largest share of *Asian messenger* users. The *CPU type* correlations shown in Figure 9 (bottom) indicate this user population is more likely to purchase middle-of-the-road systems equiped with i5 CPUs.

Facebook Messenger is a popular instant messaging application in the United States and across the globe. Figure 10 shows correlation of the *Facebook Messenger* proto with *geolocation* and *system type*. Latin America and Brazil show increased likelihood of using Facebook Messenger. On the other hand, Facebook has not penetrated well into Russia to date. Russia has its own competing social media network, VKontakte, with more than six times the number of users than Facebook has in the country. Our analysis shows Russian users are less likely to transition into the *Facebook Messenger* proto state. With regards to *system type*, *Facebook Messenger* users are more likely to use Everyday systems, as opposed to performance systems in the Premium, Multimedia, and All-in-One categories.

The *File transfers* proto represents users who spend a lot

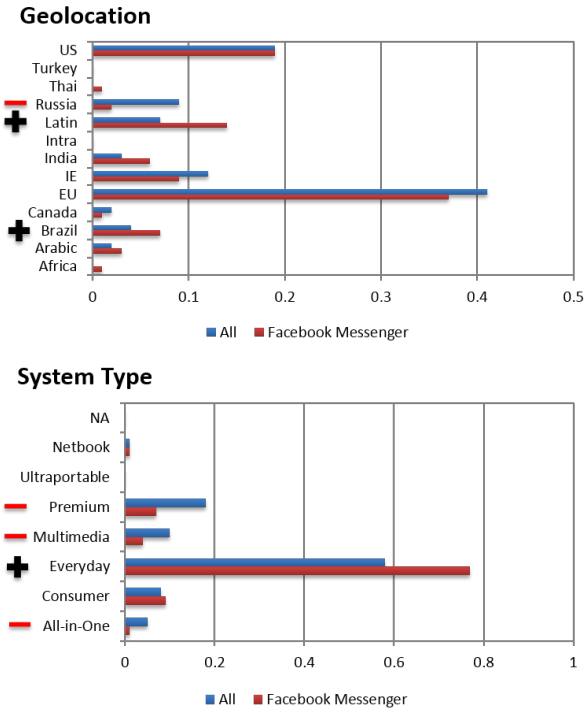

Fig. 10. *Facebook Messenger* correlation with *geolocation* and *system type*.

of CPU cycles transfering content via peer-to-peer file sharing services. An analysis of the *geolocation* side information with regards to this proto, displayed in Figure 11 (top), reveals a high likelihood of Russian users to engage in this activity, while users in the US, Latin America, and India are less likely to. The same behavior is observed in the *Skype* $\Rightarrow$ *File transfers* transition (bottom figure), where Russian users have a much higher probability of using file transfer services. IE users are also less likely to transition to *File transfers*. This is explained by an increased likelihood (not shown here) for IE users to transition to *Asian media downloads* instead, which focuses on Asian-specific peer-to-peer file sharing and video-on-demand services.

E. Purchase habit evolution

`Orion` is not restricted to analyzing PC usage information, but can work with any type of similar multivariate time series data. In this experiment, we used `Orion` to trace purchase habit evolution for 930 customers at an online grocery store. Our dataset consists of 57K online customer orders, in which 2M products were purchased. For each online customer order, we combined purchases for items that were within the same one of 905 product categories provided by the store, resulting in 1.3M overall transactions.

Figure 12 shows the first three levels of proto evolutions resulting from our experiment, in which we used parameters $p = 10$ protos, $\alpha = 5$ minimum segment length, and assigned a penalty $\beta = 0.002$ for creating new segments. Unlike the PC utilization data, purchase values (a_k) were not log-scaled. We observe that different groups of users exhibit

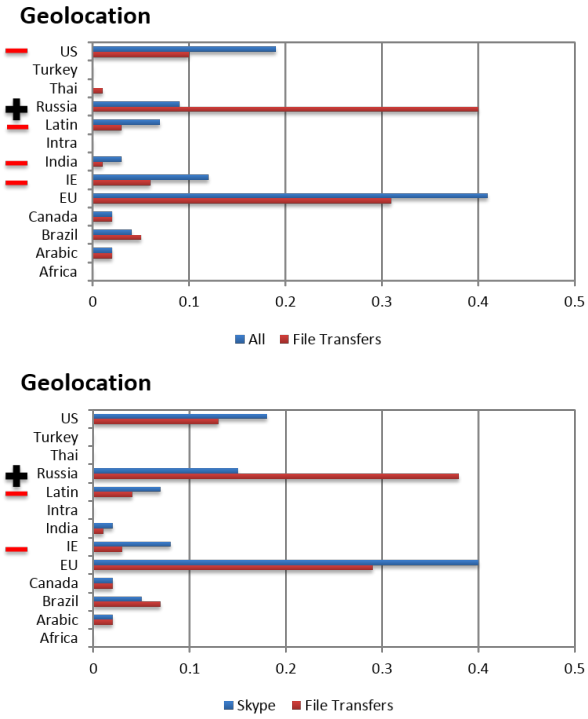

Fig. 11. *File transfers* correlation with *geolocation*.

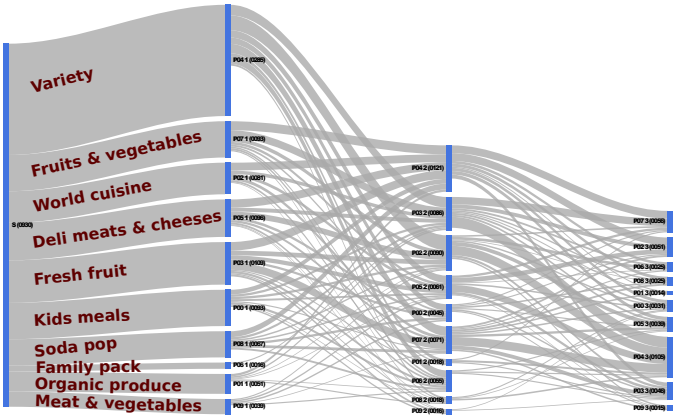

Fig. 12. Purchase habit proto evolution. See Figure 5 for information on reading this type of chart.

different purchase patterns; more than 62% of the users change patterns over time, and nearly 44% of them undergo multiple changes. Moreover, the features with high intensity in each proto are indicative of diverse purchase objectives. We present a summary of the ten protos below by listing a few of the high intensity item categories in each proto.

- $P0$ *Kids meals*. Kids cereal, Yoplait, crackers/Nabisco, ice cream/Kemps, hot dogs & wieners, mac & cheese.
- $P1$ *Organic produce*. Organic vegetables, organic bagged salads, organic fruit, organic entrees, organic cereal, organic breads.
- $P2$ *Wold cuisine*. World cuisine/Mexican fresh produce, world cuisine/Asian produce, tortilla chips, tortillas.
- $P3$ *Fresh fruit*. Berries and cherries, grapes, apples, Yoplait, melons.
- $P4$ *Variety*. Spread/butter, refrigerated juices/orange, chicken/whole & pieces, wheat bread, entree/Lean Cuisine.
- $P5$ *Deli meats & cheeses*. Deli meats/poultry, deli meats/ham, dairy/sliced cheese, deli meats/sausage.
- $P6$ *Family pack*. Family packs/beverages/water, family packs/household.
- $P7$ *Fruits & vegetables*. Grapes, apples, berries and cherries, tomatoes, pears, lettuce/greens, peppers, carrots, mushrooms.
- $P8$ *Soda pop*. Beverages/soda/pop, juices/orange/citrus.
- $P9$ *Meat & vegetables*. Chicken/whole & pieces, apples, grapes, bread/wheat, vegetables/cooking vegetables, peppers, tomatoes.

There are certain purchase bahaviors that are more stable than others. For example, the majority of users in *Family pack* ($P6$) continue to purchase items in bulk. Similarly, customers who primarily purchase *Organic produce* or *Kids meals* are likely to continue the habit. On the other hand, *Variety* represents customers with well-balanced purchase baskets. The proto does not contain any item with high intensity. While half of the customers that start in this state remain there, many transition to other, slightly unbalanced purchase habits.

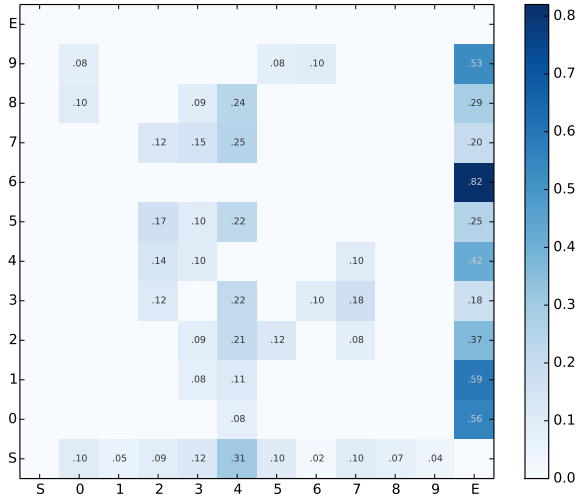

Fig. 13. Purchase habit proto transitions. See Figure 6 for information on reading this type of chart.

The proto-to-proto transition probability matrix displayed in Figure 13 confirms our observation from the proto evolutions. Protos with high *End* state probablity (rows with high values in column E) denote stable protos and include *Family pack*, *Kids meals*, *Organic produce*, and *Meat & vegetables*. *Soda pop* and *Organic produce* seem to be "early" points, as they have almost no fan-in (empty columns 8 and 9), but fairly high fan-out. While *Variety* is fairly stable, it is also a transition point into other purchase habit states such as *Fruits & vegetables*, *Soda pop*, *Deli meats & cheeses*, and *Fresh fruit* (relatively high scores for these protos' rows in column 4).

Overall, `Orion` was able to identify categories of users with diverse purchase habits and characterize their purchase habit evolution. The characterization does not come at a loss of data resolution. `Orion` produces intelligible results, identifying both interesting customer purchase behaviors and common purchase habit transitions.

TABLE III
ORION EXECUTION TIMES FOR INCREASING NUMBERS OF PROTOS

dataset / p	10	20	30	40	50
PCB	63.87	118.66	173.51	229.33	283.85
grocery	7.66	14.58	21.27	28.10	34.95

F. Efficiency

Orion was designed to be very efficient. It incrementally computes the squared error between protos and sequence vectors and is thus able to scale linearly in the number of protos. In Table III, for the two datasets we have experimented with, we present Orion execution times for $p \in \{10, 20, \ldots, 50\}$. We use the same β and α parameters as are used in our previous experiments for each dataset, respectively. Experiments were executed on an Intel i7 desktop computer, with 8Gb RAM, using a single core [2].

VI. Conclusion and Future Work

In this work, we formulated the problem of characterizing resource usage evolution as a cross-usage multivariate time series segmentation and developed a dynamic programming algorithm to find the optimal solution. Our proposed algorithm, Orion, iterates between finding optimal segmentations of user sequences via dynamic programming and deriving prototipical usage vectors (*protos*) from the segmentations.

Orion was primarily developed for and evaluated on the task of understanding and characterizing computer usage evolution. This has represented a challenging test bed, due to the very high dimensionality of the application space, along with incomplete and sometimes innacurate classification of application executables. Nevertheless, Orion was shown effective both in detecting usage patterns shared by many users and tracking the behavioral evolution of users through time. The discovered usage behaviors were generally found to be coherent with side information about the users' systems and locations. Furthermore, we demonstrated Orion's versatility through a study on the evolution of purchase patterns at an online grocery store.

There are several possible future directions for this work. From a data preparation and representation perspective, Orion could be extended to model sub-application categories, by using dimensionality reduction techniques to generate rich yet more concise protos. Our method could also be extended by generalizing some of the assumptions we placed on the segment's properties; particularly, instead of assuming that the usage in each segment is constant, we may investigate if the usage can be predicted based on previous within-segment behavior.

Acknowledgment: This work was supported in part by NSF (IIS-0905220, OCI-1048018, CNS-1162405, IIS-1247632, IIP-1414153, IIS-1447788), Army Research Office (W911NF-14-1-0316), Intel Software and Services Group, and the Digital Technology Center at the University of Minnesota. Access to research and computing facilities was provided by the Digital Technology Center and the Minnesota Supercomputing Institute.

[2] Orion source code and sample randomly generated data are available at http://cs.umn.edu/~dragos/orion.

References

[1] E. Terzi and P. Tsaparas, "Efficient algorithms for sequence segmentation," in *Proc. SIAM Int. Conf. on Data Mining (SDM)*, 2006, pp. 314–325.

[2] J. Himberg, K. Korpiaho, H. Mannila, J. Tikanmäki, and H. Toivonen, "Time Series Segmentation for Context Recognition in Mobile Devices," in *Proc. IEEE Int. Conf. on Data Mining (ICDM)*, 2001, pp. 203–210.

[3] J. Abonyi, B. Feil, S. Nemeth, and P. Arva, "Modified Gath-Geva clustering for fuzzy segmentation of multivariate time-series," *Fuzzy Sets and Systems*, vol. 149, pp. 39–56, 2005.

[4] D. Kulic and Y. Nakamura, "Scaffolding on-line segmentation of full body human motion patterns," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2008, pp. 2860–2866.

[5] F. Chamroukhi, S. Mohammed, D. Trabelsi, L. Oukhellou, and Y. Amirat, "Joint segmentation of multivariate time series with hidden process regression for human activity recognition," *Neurocomputing*, vol. 120, pp. 633–644, 2013.

[6] E. Lebarbier, "Detecting multiple change-points in the mean of Gaussian process by model selection," *Signal Processing*, vol. 85, no. 4, pp. 717–736, 2005.

[7] N. Dobigeon, J.-Y. Tourneret, and J. D. Scargle, "Joint Segmentation of Multivariate Astronomical Time Series: Bayesian Sampling With a Hierarchical Model," *IEEE Trans. on Signal Processing*, vol. 55, no. 2, pp. 414–423, 2007.

[8] F. Duchene, C. Garbay, and V. Rialle, "Learning recurrent behaviors from heterogeneous multivariate time-series," *Artificial Intelligence in Medicine*, vol. 39, pp. 25–47, 2007.

[9] N. Omranian, S. Klie, B. Mueller-Roeber, and Z. Nikoloski, "Network-based segmentation of biological multivariate time series," *PLoS ONE*, vol. 8, no. 5, p. e62974, 2013.

[10] D. Trabelsi, S. Mohammed, F. Chamrouki, L. Oukhellou, and Y. Amirat, "Activity recognition using Hidden Markov Models," in *Proc. Workshop on New and Emerging Technologies in Assistive Robotics, in conj. with IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2011.

[11] M. Ramoni, F. Sebastiani, and P. Cohen, "Bayesian clustering by dynamics," *Machine Learning*, vol. 47, pp. 91–121, 2002.

[12] K. Yang and C. Shahabi, "An efficient k-nearest-neighbor search for multivariate time series," *Information and Computation*, vol. 205, pp. 65–98, 2007.

[13] N. Wang, X. Liu, and J. Yin, "Improved gath-geva clustering for fuzzy segmentation of hydrometeorological time series," *Stochastic Environmental Research and Risk Assessment*, vol. 26, pp. 139–155, 2012.

[14] R. Bellman and R. Roth, "Curve fitting by segmented straight lines," *Journal of the American Statistical Society*, vol. 64, pp. 1079–1084, 1969.

[15] I. Auger and C. Lawrence, "Algorithms for the optimal identification of segment neighborhoods," *Bull. of Math. Biology*, vol. 51, pp. 39–54, 1989.

[16] B. Jackson, J. D. Scargle, D. Barnes, S. Arabhi, A. Alt, P. Gioumousis, E. Gwin, P. Sangtrakulcharoen, L. Tan, and T. T. Tsai, "An algorithm for optimal partitioning of data on an interval," *IEEE Signal Processing Letters*, vol. 12, pp. 105–108, 2005.

[17] H. Guo, X. Liu, and L. Song, "Dynamic programming approach for segmentation of multivariate time series," *Stochastic Environmental Research and Risk Assessment*, pp. 1–9, 2014.

[18] S. K. Tyler, S. Pandey, E. Gabrilovich, and V. Josifovski, "Retrieval models for audience selection in display advertising," in *Proc. ACM Conf. on Information and Knowledge Management (CIKM)*, 2011, pp. 593–598.

[19] M. Aly, A. O. Hatch, V. Josifovski, and V. K. Narayanan, "Web-scale user modeling for targeting," in *Proc. ACM Conf. on World Wide Web (WWW)*, 2012, pp. 3–12.

[20] M. Aly, S. Pandey, V. Josifovski, and K. Punera, "Towards a Robust Modeling of Temporal Interest Change Patterns for Behavioral Targeting," in *Proc. ACM Conf. on World Wide Web (WWW)*, 2013, pp. 71–81.

[21] A. Kehagias, E. Nidelkou, and V. Petridis, "A dynamic programming segmentation procedure for hydrological and environmental time series," *Stochastic Environmental Research and Risk Assessment*, vol. 20, no. 1-2, pp. 77–94, 2006.