# An Effective Approach for Mining Mobile User Habits

Huanhuan Cao[1,2] Tengfei Bao[1,2] Qiang Yang[3] Enhong Chen[1] Jilei Tian[2]
[1]University of Science and Technology of China [2]Nokia Research Center
[3]Hong Kong University of Science and Technology
[1]{happia.cao, jilei.tian}@nokia.com [2]{tfbao92, cheneh}@ustc.edu.cn
[3]qyang@cse.ust.hk

## ABSTRACT

The user interaction with the mobile device plays an important role in user habit understanding. In this paper, we propose to mine the associations between user interactions and contexts captured by mobile devices, or *behavior patterns* for short, from context logs to characterize the habits of mobile users. The extensive experiments on the collected real life data clearly validate the ability of our approach for mining effective behavior patterns.

## Categories and Subject Descriptors

H.2.8[Database Management] [**Database Applications**]: Data Mining

## General Terms

Algorithms, Experimentation

## Keywords

Mobile user, habit mining, behavior patterns.

## 1. INTRODUCTION

The rich user interaction information captured by the mobile device can be used to understand user habits, which can bring a great business value, such as targeted advertising and personalized recommendation. A distinct property of the user interactions with mobile devices is that they are usually associated with volatile contexts, such as waiting a bus, driving a car, or doing shopping. Intuitively, some user interactions are context-aware, that is, the occurrences of these user interactions are influenced by the contexts of users. For example, some users would like to listen to music with their smart phones when taking a bus to the workplace but rarely do the same thing on other contexts. Therefore, we argue that the associations between user interaction records and the corresponding contexts, which are refereed as *behavior patterns*, can be used to characterize user habits.


*CIKM'10,* October 26–30, 2010, Toronto, Ontario, Canada.


Context logs collect the history context data and interaction records of mobile users, and thus can be used as data sources for mining behavior patterns. However, mining behavior patterns is not a trivial problem because it cannot be addressed by the traditional association rule mining [1]. To this end, we propose an effective approach for behavior pattern mining which takes context logs as time ordered sequences of context records and calculates the support of a context by taking into account its time ranges of appearances. Experimental results on real data sets clearly show that our approach outperforms the traditional association rule mining approach for mining behavior patterns.

## 2. PROBLEM STATEMENT

To ease presenting the problem of behavior pattern mining, we first define some related notions as follows.

DEFINITION 1 (CONTEXT). *Given a contextual feature set $F = \{f_1, f_2, ..., f_K\}$, a* ***context*** *$C_i$ is a group of contextual feature-value pairs, i.e., $C_i = \{(x_1 : v_1), (x_2 : v_2), ..., (x_l : v_l)\}$, where $x_n \in F$ and $v_n$ is the value for $x_n$ $(1 \leq n \leq l)$. A context with $l$ contextual feature-value pairs is called an* ***$l$-context****.*

DEFINITION 2 (SUB-CONTEXT, SUPER-CONTEXT). *Given two contexts $C_i$ and $C_j$, if $\forall_{p_i \in C_j} p_i \in C_i$, where $p_i$ denotes a contextual feature-value pair, $C_j$ is called a* ***sub-context*** *of $C_i$ and $C_i$ is called a* ***super-context*** *of $C_j$.*

A contextual feature denotes a type of context data, such as day period, location, audio level, etc. For simplicity of operating contexts, such as context comparison, we require that contextual feature-value pairs be sorted in a predefined order of contextual features.

DEFINITION 3 (INTERACTION RECORD). *An* ***interaction record*** *is an item in the* ***interaction set*** *$\Gamma = \{I_1, I_2, ..., I_Q\}$, where $I_n (1 \leq n \leq Q)$ denotes a kind of user interaction.*

Interaction records capture the occurrences of user interactions with mobile devices, such as listening to music, message session or Web browsing.

DEFINITION 4 (CONTEXT RECORD, CONTEXT LOG). *A* ***context record*** *$r =< Tid, C_i, I >$ is a triple of a timestamp $Tid$, a context $C_i$, and a user interaction record $I$. A* ***context log*** *$R = r_1 r_2 ... r_{N_R}$ is a group of context records ordered by timestamps.*

A context record captures the most detailed *available* context and the occurrence of a user interaction during a time interval. We mention "available" because a context record may miss the values of some contextual features though the set of context-features whose values should be collected is predefined. Moreover, interaction records can be empty (denoted as "Null") if no interaction happen during the time interval.

Context logs integrate the history context data and interaction records of mobile users, and thus can be data sources for mining behavior patterns. However, mining behavior patterns is not a trivial problem because the traditional association rule mining approach suffers the unbalanced occurrences of contexts and interaction records. For example, suppose that Sam usually listens to rock music when taking a bus during workdays' AM8:00-9:00. When the context {(*Is a holiday?: No*),(*Time range: AM8:00-9:00*),(*Transportation: On vehicle*)} appears, Sam usually listens to rock music but the exact time points when the interaction happens are uncertain. Consequently, the occurrences of the interaction *Listening to rock music* are very sparse compared with the occurrences of the context {(*Is a holiday?: No*),(*Time range: AM8:00-9:00*),(*Transportation: On vehicle*)} in context records, which causes the traditional association rule mining approach can hardly discover the behavior pattern {(*Is a holiday?: No*),(*Time range: AM8:00-9:00*),(*Transportation: On vehicle*)} $\Longrightarrow$ *Listening to rock music*. One may argue for a method in which we first extract the context records which contain non-empty interaction records and then apply the traditional association rule mining. This alternative approach loses the discriminative information that how likely no interaction happens with a given context. As a result, the calculated confidence may be meaningless. The detailed explanation of the problem can be found in [2].

From context logs we observe that if a user interaction is influenced by the context $C_i$, the corresponding interaction record $I$ usually co-occurs with $C_i$ in the time ranges when $C_i$ continuously appears in several adjacent context records. Therefore, we propose to not only consider the co-occurrences of contexts and interaction records in separate context records but also consider their co-occurrences in the whole time ranges of contexts. To be specific, we take context logs as time ordered sequences of context records and calculate the support of a context by taking into account its time ranges of appearances. For a candidate behavior pattern $C_i \Longrightarrow I$, the support (denoted as $Sup(C_i \Longrightarrow I)$) is still calculated by counting the context records where $C_i$ and $I$ co-occur. But for a context $C_i$, the support (denoted as $Sup(C_i)$) is calculated separately in two different cases. If $C_i$ continuously appears in several adjacent context records and all of these context records only contain empty interaction records, we will count $C_i$ once. Otherwise, we will count $C_i$ by the number of non-empty interaction records in these context records. We discriminate the two different cases because in this way we ensure that $Sup(C_i)$ is always not smaller than $\sum_I Sup(C_i \Longrightarrow I)$, which is a basic assumption for calculating the confidence.

The formal problem statement of behavior pattern mining is as follows.

Definition 5 (Match Context). *A context record* $r = < Tid, C_i, I >$ ***match**es a context* $C_j$ *iff* $C_j$ *is a sub-context of* $C_i$. *For simplicity, we can use* $r \perp C_i$ *to denote that* $r$ *matches* $C_i$.

Definition 6 (Context Range). *Given a context* $C_i$ *and a context log* $R = r_1 r_2 ... r_{N_R}$, *we say that* $R_i^n = r_i r_{i+1} ... r_{i+n}$ *is a* ***context range*** *of* $C_i$ *iff 1)*$\forall_{1 \le j \le n}(r_{i+j} \perp C_i)$*; 2) Both* $(r_{i-1} \perp C_i)$ *and* $(r_{i+n+1} \perp C_i)$ *are false.*

Definition 7 (Support, Confidence). *Given a context* $C_i$,*an interaction record* $I$, *and a context log* $R$, *the* ***support*** *of* $C_i$ *w.r.t.* $I$ $(Sup(C_i \Longrightarrow I))$ *is* $\sum_m Count_m(I)$, *where* $Count_m(I)$ *denotes the occurrence number of* $I$ *in the* $m$*-th context range of* $C_i$.

*Given an interaction set* $\Gamma$, *the* ***support*** *of* $C_i$ $(Sup(C_i))$ *is* $\sum_{I \in \Gamma} Sup(C_i \Longrightarrow I) + N_0$, *where* $N_0$ *denotes the number of context ranges of* $C_i$ *that do not contain any non-empty interaction record. Moreover, the* ***confidence*** *of* $C_i$ *w.r.t.* $I$ $(Conf(C_i \Longrightarrow I))$ *is* $\frac{Sup(C_i \Longrightarrow I)}{Sup(C_i)}$.

Definition 8 (Promising Context, Behavior Pattern). *Given a context* $C_i$, *a context log* $R$, *two user defined parameters* ***min_sup*** *and* ***min_conf***, *if* $\exists_I Sup(C_i \Longrightarrow I) \ge min\_sup$, $C_i$ *is called a* ***promising context***. *Moreover, if* $Sup(C_i \Longrightarrow I) \ge min\_sup$ *and* $Conf(C_i \Longrightarrow I) \ge min\_conf$, $C_i \Longrightarrow I$ *is called a* ***behavior pattern***.

# 3. ALGORITHM FOR BEHAVIOR PATTERN MINING

Most of the traditional association rule mining algorithms divide the mining procedure into two stages. In the first stage, all frequent itemsets are found from the transaction data base. In the second stage, the rules are generated from the frequent itemsets and their confidences are calculated. This strategy may significantly reduce the total memory requirement since the number of association rules may increase exponentially with the number of frequent items. For example, given a frequent itemset $abc$, the possible association rules are $ab \Longrightarrow c$, $ac \Longrightarrow b$, $bc \Longrightarrow a$, $a \Longrightarrow bc$, $b \Longrightarrow ac$, and $c \Longrightarrow ab$. However, in behavior pattern mining, we can integrate the two stages because the upper bound of the number of behavior patterns is linear to the number of promising contexts.

A naive algorithm for behavior pattern mining enumerates all contexts appeared in the context log as candidate promising contexts and then counts their supports and confidences w.r.t. each interaction by scanning the context log. However, this algorithm is inefficient since its time complexity is $O(N_R \cdot \prod_{k=1}^{K} N_k)$, where $N_R$ indicates the number of context records and $N_k$ indicates the number of values for the contextual feature $f_k$ appeared in the context log. Wisely, we can reduce the number of candidate promising contexts $\prod_{k=1}^{K} N_k$ to a much smaller number by leveraging the *Apriori-property* of behavior patterns. That is, given a context $C_i$ and an interaction record $I$, if $Sup(C_i \Longrightarrow I) > \alpha$, for any sub-context of $C_i$ denoted as $C_j$, we can conclude that $Sup(C_j \Longrightarrow I) > \alpha$. It means that if a context $C_i$ is not a promising context, i.e., $!\exists_I Sup(C_i \Longrightarrow I) \ge min\_sup$, none of $C_i$'s super-contexts is promising. Along this line, we propose an algorithm for behavior pattern mining based on the framework of Apriori-all algorithm [1], which is named

GCPM (**G**enerating **C**andidate promising contexts for behavior **P**attern **M**ining). The main idea of GCPM is to generate candidate promising $l+1$-contexts by *joining* promising $l$-contexts. The notion of joining contexts is defined as follows.

DEFINITION 9 (JOIN CONTEXT). *Given two contexts* $C_i = \{(x_1 : v_1), (x_2 : v_2), ..., (x_l : v_l)\}$ *and* $C_j = \{(y_1 : u_1), (y_2 : u_2), ..., (y_l : u_l)\}$, *if* $\forall_{1<n\leq l}(x_n = y_{n-1} \wedge v_n = u_{n-1})$, *we say that* $C_i$ *and* $C_j$ *can join. The joined context of* $C_i$ *and* $C_j$ *is denoted as* $C_i \cdot C_j = \{(x_1 : v_1), (x_2 : v_2), ..., (x_l : v_l), (y_l : u_l)\}$.

Given a context log $R$ and a user interaction set $\Gamma = \{I_1, I_2, ... , I_Q\}$, GCPM first enumerates all 1-contexts appeared in $R$ as $\Lambda^1 = \{C_i^1\}$ and set $l$ to be 1. Then, given a set of candidate promising $l$-contexts $\Lambda^l$, the following procedure is executed iteratively until no candidate promising contexts can be generated.

1) Scan $R$ and find all promising $l$-contexts from $\Lambda^l$ and the corresponding behavior patterns.

2) Generate $\Lambda^{l+1}$ by joining candidate promising $l$-contexts $C_i^l$ and $C_j^l$ where $\exists_{I\in\Gamma} Sup(C_i^l, I) \geq min\_sup \wedge Sup(C_j^l, I) \geq min\_sup$;

3) If $\Lambda^{l+1} = \phi$, the algorithm terminates. Otherwise $l++$ and go to 1).

Through iterative joining stages, GCPM largely reduces the number of candidate promising contexts from $\prod_{k=1}^{K} N_k$ to $\sum_l |\Lambda^l|$, which is much smaller than the former number in practice. Therefore, the time complexity of GCPM is $O(N_R \cdot \sum_l |\Lambda^l|)$.

The main cost of GCPM comes from counting the supports of candidate promising contexts w.r.t. each interaction. To improve the efficiency of this stage, we introduce a novel data structure called *CH-Tree (**C**ontext **H**ash **Tree**)* for quickly updating the supports of candidate promising contexts w.r.t. each interaction when scanning context records. It is worth noting that Park et al. [4] proposed a hash based algorithm for association rule mining which can largely reduce the number of candidate 2-frequent items. The objective of their approach is different from ours because CH-Trees are designed for speeding up the support calculation but not for reducing the number of candidate promising contexts. Actually, the latter problem has been addressed by the joining stage of GCPM.

A CH-Tree has a tree-like representation of the nodes as follows.

- **Root Node(RN)**: each CH-Tree has one root node. This node contains $K$ pointers that point to *intermediate nodes* or null, where $K$ is the total number of contextual features in a given contextual feature set.
- **Intermediate Node(IN)**: each intermediate node contains $H$ pointers that point to *leaf nodes* or null, where $H$ is the output range of a predefined hash function $Hash()$.
- **Leaf Node(LN)**: each leaf node contains a group of contexts. Each context $C_i$ is associated with array of support counters $C_i.sup[]$, a boolean tag $C_i.hasInter$ to indicates whether the nearest context range of $C_i$ contains at leat one non-empty interaction record, and a tag $C_i.isMatch$ to indicates whether $C_i$ is matched by the current context record $r$.

Initially, an empty CH-Tree only has a root node. The intermediate nodes and leaf nodes are created during the process of inserting candidate promising contexts. The operation $Insert()$ is used to insert candidate promising contexts into a CH-Tree. Given a received context $C_i$, the operation $Insert()$ first selects a contextual feature $f_k$ and assigns $C_i$ to the intermediate node pointed by the $k$-th pointer of the root node, denoted as $IN_k$. If $IN_k$ does not exist, the operation $Insert()$ will create it first. Then it puts $C_i$ into the leaf node pointed by the $h$-th pointer of $IN_k$, where $h = Hash(v_k)$ and $v_k$ indicates the value of $f_k$ in $C_i$. Similarly, if the leaf node does not exist, the operation $Insert()$ will also create it first.

After all candidate promising contexts are inserted, the CH-Tree is used to calculate the supports of the candidate promising contexts in it when scanning a context log $R$. Given a context record $r$, the operation $Count()$ assign $r$ to each existing intermediate nodes. When the intermediate node $IN_k$ receives $r$, it returns the leaf node pointed by its $h$-th pointer as a leaf node that may contain contexts matched by $r$, where $h = Hash(u_k)$ and $u_k$ indicates the value of $f_k$ in $r$. Only the candidate promising contexts in these returned leaf nodes are checked for updating their supports. In contrast, in the original GCPM algorithm all candidate promising contexts are checked whether their supports should be updated for $r$. Therefore, the efficiency of updating supports of contexts is improved by CH-Trees.

To reduce the average number of checked candidate promising contexts, it is desirable to partition the contexts to each leaf node as equally as possible. To this end, we expect that the value distribution for the selected hash key $f_k$ is as sparse as possible. Therefore, the operation $Insert()$ selects the $f_k$ in a sparseness descending order. The sparseness of the value distribution for $f_k$ can be evaluated by the information entropy, i.e., $H(f_k) = -\sum_{v_k} P(v_k|f_k) log P(v_k|f_k)$, where $P(v_k|f_k)$ denotes the probability that the value of $f_k$ is $v_k$. $P(v_k|f_k)$ can be estimated as $\frac{Freq(f_k,v_k)}{Freq(f_k)}$, where $Freq(f_k, v_k)$ indicates the frequency of contextual feature-value pair $(f_k : v_k)$, and $Freq(f_k)$ indicates the frequency of contextual feature $f_k$.

# 4. EXPERIMENTS

To evaluate the proposed approach, we conduct extensive experiments on real context logs of mobile users. We build a data collection system for collecting the rich context data such as GPS data, system information, GSM data, call log, sensor data, and interaction records of 50 college volunteers spanning for one month. Due to the page limitation, we only show the detailed experimental results for two randomly selected volunteers' context logs, namely, data set $D_A$ and data set $D_B$. But we still report the general phenomenon shown in the experiments for other context logs.

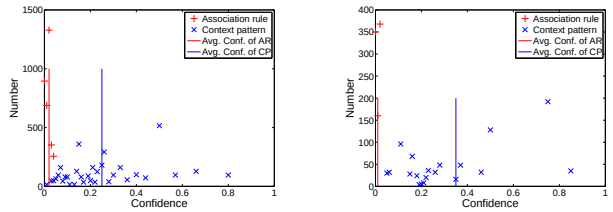
To evaluate the ability of our approach for mining behavior patterns, we use the association rule mining approach as the baseline method. Notice that in the following paragraph we use "association rules" only to denote the association rules whose antecedents are contexts and the consequents are interaction records. The implementation of the association rule mining approach is based on the FP-Growth algorithm [3].

Figure 1 and Figure 2 compare the distribution of behavior patterns (CP) and that of association rules (AR) with

**Table 1: Examples of mined behavior patterns.**

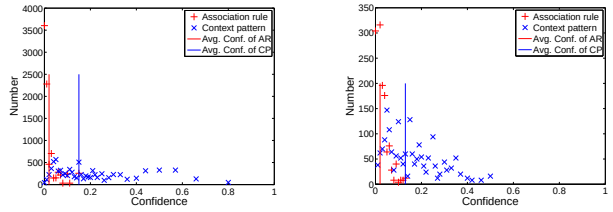
| | behavior pattern | *Sup* | *Conf* |
|---|---|---|---|
| $D_A$ | Context: {(Is a holiday?: No),(Day period: Morning),(Time range: AM8:00-9:00),(Speed: High(>20km/h))} $\Longrightarrow$ Interaction: Listening to music | 14 | 0.64 |
| | Context: {(Is a holiday?: No),(Day period: Afternoon),(Profile: General),(Movement: Not moving), (Coordinate: (39.8554 116.4097))} $\Longrightarrow$ Interaction: Message session | 15 | 0.68 |
| $D_B$ | Context: {(Is a holiday?: Yes)(Time range: AM7:00-8:00),(Battery level: High(50%-80%)),(Ring type: Silent)} $\Longrightarrow$ Interaction: Listening to visible radio | 5 | 0.62 |
| | Context: {(Day name: Thursday),(Day period: Noon),(Time range: AM12:00-PM13:00),(Profile: Silent)} $\Longrightarrow$ Interaction: Accessing Web | 4 | 1 |

respect to confidence in $D_A$ and $D_B$, respectively. In these figures, the confidences are accurate to 0.01. To observe the difference of two distributions more clearly, we mark the average confidences (Avg. Conf.) of each distribution. From these figures, we can see that the confidences of association rules are usually too low to let us distinguish meaningful rules from noisy data. The experiments on other context logs show the similar phenomenon.




(a) $min\_sup = 4$ (b) $min\_sup = 6$

**Figure 1: The distributions of behavior patterns and association rules in $D_A$.**




(a) $min\_sup = 4$ (b) $min\_sup = 6$

**Figure 2: The distributions of behavior patterns and association rules in $D_B$.**

We manually check the mined behavior patterns and find that most of them can reflect the habits of users. Table 1 shows some behavior patterns mined from $D_A$ and $D_B$. These behavior patterns reflect some habits of the corresponding users. For example, the first pattern of $D_A$ implies that the corresponding volunteer usually listens to music during workdays' AM8:00-9:00 when taking a vehicle (Speed: High(>20km/h)).

To achieve a more objective evaluation, we also ask volunteers to evaluate the behavior patterns mined from their own context logs. Precisely, for each volunteer, we mine all behavior patterns with $min\_sup$=2 and $min\_conf$=0.5 from his (or her) context log. Then we select at most top 20 behavior patterns for each interaction instead of using all the mined behavior patterns. It's because the total number of the mined behavior patterns usually exceeds to several hundreds and thus it may bring too much burden to evaluate all behavior patterns. Given a behavior pattern to be evaluated, the volunteer can select one from the following three remarks:

- I: it is **I**nteresting. It's correct but I have not realized till now.
- Y: **Y**es, it is correct. I usually follow this pattern and I know.
- N: **N**o, it is not correct. I rarely follow this pattern.

To ensure the quality of the evaluation, we generate a copy for each behavior pattern and randomly mix them with the original ones. If a behavior pattern is assigned different remarks from that of its copy, we will revisit it again. The evaluation result shows that all volunteers gave more than 95% positive remarks (I+Y) for the mined behavior patterns and the average ratio of positive remarks for each volunteer is more than 98%.

We also evaluate the efficiency of the proposed algorithms including GCPM and its optimization, denoted as GCPM-H, for $D_A$ and $D_B$. Both algorithms are implemented with standard C++ on a 2×2.0G CPU, 2G main memory PC. The experiments on all context logs show that the average running time of GCPM-H is about 10% of that of GCPM.

## 5. CONCLUSION

In this paper, we propose an effective approach for mining behavior patterns which takes context logs as time ordered sequences of context records and takes into account the co-occurrences of contexts and interaction records in the whole time ranges of contexts. The experiments on real data sets clearly show that our approach is effective, efficient and promising.

**Acknowledgement.** This work is supported by grants from Natural Science Foundation of China (No.60775037), Key Program of National Natural Science Foundation of China (No.60933013) and Nokia.

## 6. REFERENCES

[1] Agrawal, R. and Srikant, R. Fast algorithms for mining association rules. In *VLDB'94*, pages 487–499, 1994.

[2] Cao, H., Bao, T., and Yang, Q. et al. An effective approach for mining mobile user habits. Technical report, 2009. http://dm.ustc.edu.cn/paperlist.html

[3] Han, J., Pei, J., and Yin, Y. Mining frequent patterns without candidate generation. In *SIGMOD'00*, pages 1–12. ACM, 2000.

[4] Park, J. S., Chen, M., and Yu, P. S. An effective hash-based algorithm for mining association rules. In *SIGMOD '95*, pages 175–186, 1995.