4-2006

SGPM: Static Group Pattern Mining using apriori-like sliding window

John GOH

David Taniar

Ee Peng LIM
Singapore Management University, eplim@smu.edu.sg

DOI: https://doi.org/10.1007/11731139_48

Citation

GOH, John; Taniar, David; and LIM, Ee Peng. SGPM: Static Group Pattern Mining using apriori-like sliding window. (2006). *Advances in Knowledge Discovery and Data Mining: 10th Pacific-Asia Conference, PAKDD 2006, Singapore, April 9-12: Proceedings*. 3918, 415-424. Research Collection School Of Information Systems.
Available at: https://ink.library.smu.edu.sg/sis_research/895

SGPM: Static Group Pattern Mining Using Apriori-Like Sliding Window

John Goh[1], David Taniar[1], and Ee-Peng Lim[2]

[1] Monash University, Clayton, Vic 3800, Australia
{Jen.Goh, David.Taniar}@infotech.monash.edu.au
[2] Nanyang Technological University, Singapore 639798, Singapore
aseplim@ntu.edu.sg

Abstract. Mobile user data mining is a field that focuses on extracting interesting pattern and knowledge out from data generated by mobile users. Group pattern is a type of mobile user data mining method. In group pattern mining, group patterns from a given user movement database is found based on spatio-temporal distances. In this paper, we propose an improvement of efficiency using area method for locating mobile users and using *sliding window* for *static group pattern mining*. This reduces the complexity of valid group pattern mining problem. We support the use of static method, which uses areas and *sliding windows* instead to find group patterns thus reducing the complexity of the mining problem.

1 Introduction

Modern society is increasingly adopting mobile phones [15]. Mobile phone is increasing complex, and providing more user oriented services to mobile users and thus is becoming more and more beneficial to have a mobile phone [16]. Mobile phones are usually carried by a single user, and are personalized to that particular mobile user. As mobile phones now can be personalized and tracked [3, 14], it opens up a new dimension of data mining, called mobile user data mining [5, 17, 18], in which interesting knowledge can be mined from the record of the mobile user's background, places visited, and details of the places visited.

Data mining focuses on methods and algorithms in order to extract interesting patterns and knowledge from mobile users. Data mining have since been applied into different areas such as temporal domain [4, 7, 12, 13], spatial temporal domain [10, 11], and market basket analysis domain such as association rules [1, 8, 9] and sequential patterns [2].

Group pattern [17, 18] developed by Wang et al. is useful in determining grouping information over a large geographical location, a large number of mobile users and over a large duration of time series through data mining. However, one major limitation of group pattern is that it uses Euclidean distance to determine the relative proximity among mobile users. This is a method which becomes a limitation when the size of total number of mobile users through the time horizon becomes large, leading to complex dataset and reduced efficiency. The rationale behind group pattern is such

W.K. Ng, M. Kitsuregawa, and J. Li (Eds.): PAKDD 2006, LNAI 3918, pp. 415 – 424, 2006.

that human beings physically close together over a certain time occurring frequently can be deemed as close socially [6].

In real life mobile environment there are obstacles, which will be termed as static objects for the rest of this paper. These static objects are such as things that do not move in the mobile environment. For example, walls, doors, phone booths, floors are all static objects. As group pattern uses Eucilidean distance, or direct distance between two mobile users in order to determine their social proximity, the weakness is that if two mobile users is separated by a wall (i.e. between two classroom), they will be deemed to be as a close group. The result of this is that there will be more group pattern generated in the end of the process than the true number of group pattern there really is. This is because people separated by a wall are principally not close together.

2 Background

Data source for group pattern [17, 18] mining is a user movement database defined by $D = (D_1, D_2, \ldots, D_M)$, where D_i is a time series containing tuples $(t, (x, y, z))$ denoting the (x, y, z) values respectively of user u_i at time point t. For conformance to previous definition, we denote the location of a user u_i at time t by $u_i[t].p$ and his/her (x, y, z) values at time t by $u_i[t].x$, $u_i[t].y$, $u_i[t].z$ respectively. It is also assumed that all user locations are known at every time point and the interval between t and t+1 is fixed.

Definition 1. *Given a set of users G, a maximum distance threshold max_dis, and a minimum time duration threshold min_dur, a set of consecutive time points* $[t_a, t_b]$ *is called a* ***valid segment*** *of G, if*

1. $\forall u_i, u_j \ni G, \forall t, t_a \leq t \leq t_b, d(ui[t].p, uj[t].p) \leq max\_dis$;
2. $t_a = 0$ or $\exists u_i, u_j \ni G, d(u_i[t_a\text{-}1].p, u_j[t_a\text{-}1].p) > max\_dis$;
3. $t_b = N - 1$ or $\exists u_i, u_j \ni G, d(u_i[t_b+1].p, u_j[t_b+1].p) > max\_dis$;
4. $(t_b - t_a + 1) \geq min\_dur$;

The distance fuction, $d()$, is defined to return the Eucilidean distance between two points, i.e., $d(u_i[t].p, u_j[t].p)$ =

$$\sqrt{(u_i[t].x - u_j[t].x)^2 + (u_i[t].y - u_j[t].y)^2 + (u_i[t].z - u_j[t].z)^2}$$

Consider the user movement database in Table 1. For min_dur = 3 and max_dis = 10, [5,8] is a valid segment of the set of users, {u2, u4}.

Definition 2. *Given a database D, a group of users G, thresholds max_dis and min_dur, we say that G, max_dis and min_dur form a* ***group pattern****, denoted by P = < G, max_dis, min_dur >, if G has a valid segment.*

In the interest of space, algorithm AGP [17, 18] is not shown. Valid segments of the group pattern P are therefore the valid segments of its G component. Group pattern with k users is also known as **k-group pattern**. In a user movement database, a group pattern [17, 18] may have multiple valid-segments. The combined length of these valid segments is called the weight count of the pattern. We quantify the significance of the pattern by comparing its *weight count* with the overall time duration.

Definition 3. *Let P be a group pattern with valid segments* s_1, …, s_n, *and N denotes* the *number of time points in the database, the* **weight** *of P is defined as*:

$$weight(P) = \frac{\sum_{i=1}^{n} | s_i |}{N} \quad (1)$$

If the weight of a group pattern [17, 18] exceeds a threshold *min_wei*, we call it a **valid group pattern**, and the corresponding group of users a **valid group**. For example, considering the user movement database D in Table 1, if *min_wei* = 50%, the group pattern $P = <\{u_2, u_3, u_4\}, 10, 3>$ is a valid group pattern, since it has valid segments {[1,3], [6,8]} and its weight is $6/10 \geq 0.5$.

Definition 4. *Given a database D, thresholds max_dis, min_dur, and min_wei, the problem of finding all the valid group patterns (or simply valid groups) is known as* ***valid group (pattern) mining***.

3 Proposed Method: Static Group Pattern Mining (SGPM)

Group pattern [17, 18] mining is defined in Section 2. This proposal proposes a way of mining without using Euclidean distance. Euclidean distance is a formula to calculate the distance in a two dimensional space. The use of Eucilidean distance means more calculation, and also Eucilidean distance is prone to problems where two mobile users are separated by an obstacle, such as a wall. In this paper, we focuses on the issue of redefining group pattern mining, while the issue of obstacles has been proposed and addressed in another contribution.

First, we re-define how the data in mobile devices are collected. For each mobile device, it is assumed that the mobile device have some form of memory and global positioning system function, and internal system clock to determine the current time and location. In the previous proposed group pattern, data is collected as a stream for each and every second throughout the time. This automatically translates to a huge and immense amount of source data to be mined. Consider each mobile user generates a piece of coordinate (x, y) in the set of integer, the data keeps incrementing at all times. Data source for group pattern mining is a user movement database defined by $D = (D_1, D_2, \ldots, D_M)$, where D_i is a time series containing tuples $(t, (x, y))$ denoting the (x, y) values respectively of user u_i at time point t. For conformance to previous definition, we denote the location of a user u_i at time t by $u_i[t].p$ and his/her (x, y) values at time t by $u_i[t].x$, $u_i[t].y$ respectively. It is also assumed that all user locations are known at every time point and the interval between t and t+1 is fixed.

Assumption 1. *Given a mobile device* $\mathfrak{R}$, *it is assumed that* $\mathfrak{R}$ *is equipped with a location identification system, such as global positioning system where it could* *determine its position in earth, or otherwise determine which room the mobile device is located in a shopping mall.*

Assumption 2. *Given a mobile device $\Re$, it is assumed that $\Re$ is equipped with brief processing capability, and data recording facility. $\Re$ will roam around the mobile environment, and subsequently records down the user movement activity accordance to definition 1, and subsequently uploaded to the mobile user data mining centre when the recording facility is full, for mobile user data mining.*

Definition 5 (Location of Interest). *Given a mobile device $\Re$, duration threshold $\wp$ is defined. $\wp$ is an integer value that represents time unit, which can be second, minute or hour. It is set to a value that if a mobile user stops in a location for $\wp$ duration of time, then the mobile user has shown some interest in this particular location. If a mobile user spent more than $\wp$ in a location, that location is also known as location of interest (***LOI***).*

Definition 6 (Data Recording Conditions). *Given a mobile device $\Re$, variables t_{start}, t_{stop}, $t_{current}$, $t_{threshold}$, $v_{threshold}$, $v_{current}$ are defined. For $\Re$, in order to save processing time and storage space, user movement data is not recorded if the mobile user moving at a velocity $v_{current}$ where $v_{current} > v_{threshold}$.*

Explanation: This is because if the mobile user is travelling fast, it is unlikely that the mobile user have interest in the location, but merely travelling from one point to another. If $v_{current} < v_{threshold}$ which means that mobile user slows down or stationery for $t_{threshold}$ duration of time, then user movement recording starts, such that $t_{start} = t_{current} - t_{threshold}$. Once $\Re$ moves, where $v_{current} > v_{threshold}$ recording stops, such that $t_{stop} = t_{current}$.

Definition 7 (Movement Data Format). *Recording of user movement database will be represented in the format of: $u_x : [a_i(t_{start} : t_{stop}, \ldots, t_{start} : t_{stop}), \ldots, a_j(t_{start} : t_{stop}, \ldots, t_{start} : t_{stop})]$ where user x (u_x) visited area $a_i \ldots a_j$ where in a_i, user u_x is present for a set of ($t_{start} : t_{stop}$) duration of time, where each $t_{start} : t_{stop}$ is such that $t_{stop} - t_{start} > t_{threshold}$.*

Example: For example: $u_1 : a_1(0 : 5, 21 : 30), a_2(6 : 10), a_3(11 : 20)$, which represent user u_1 have visited area a_1 from time 0 : 5 and 21 : 30, and area a_2 from time 6 : 10 and area a_3 from time 11 : 20. This means user u_1 have visited a_1, a_2, a_3, and back to a1 in sequence. We define each of this record r_n.

Definition 8 (Valid Segment). *Given a set of mobile devices $\Re$, area A, t_{start}, t_{stop}, $t_{current}$, $t_{threshold}$, $v_{threshold}$, $v_{current}$, each record r_n is called a* ***valid segment*** *of G. We wish to remove the definition of weight in previous group pattern proposal, as weight is no longer required. Given database D, threshold $t_{threshold}$, $v_{threshold}$, area $a_{current}$, time $t_{current}$, t_{start}, t_{stop}, the problem of finding all the valid group patterns (or simply valid groups) is known as valid group (pattern) mining.*

4 SGPM Mining: Algorithm ASGP

We propose Apriori-like Static Group Pattern (*ASGP*) mining algorithm for the purpose of finding all valid group patterns. *ASGP* is an algorithm for the mining problem of Static Group Pattern Mining (*SGPM*). *ASGP* utilizes sliding window concept and also *Apriori* combination generation concept in order to mine all valid group patterns. Sliding window is a window defined by the size of t_{duration}. Let total time in the time series be t_{total}. Sliding window will starts from $t = 0$, until $t = t_{\text{total}} - t_{\text{threshold}}$. Each slide will involve the sliding window reference time $t_{\text{ref}} = t_{\text{ref}} + 1$.

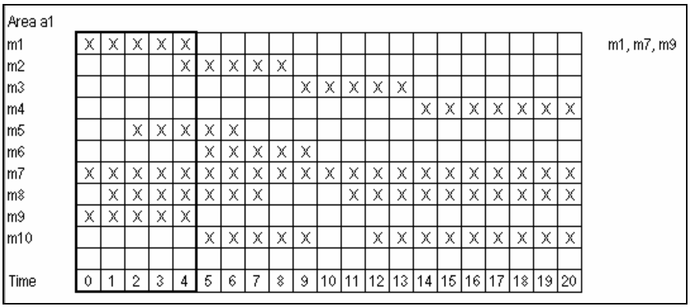

Fig. 1. Demonstration of *Sliding-Window*

Figure 1 illustrates the sliding window and the dataset in order to find all valid groups patterns. Dataset is grouped by area, which the illustration shows all mobile users (m_1, …, m_{10}) who have visited area a_1 from time (0, …, 20). For each area, (i.e. area a_1), only mobile users who have stayed in this area longer than $t_{\text{threshold}}$, is recorded through the definition in mobile devices. Sliding window is shown in $t = 0$ … $t = 4$, where it is illustrated as a highlighted border. There are altogether 10 mobile users, (m_1, …, m_{10}), and the total time ranges from $t = 0$ … $t = 20$. There are 17 passes altogether. For each pass, the sliding window will examine the mobile users in the sliding on whether they have stayed in this sliding window for the total duration of time (i.e. mobile user must stay from $t = 0$ to $t = 4$ in this window to be recorded). Illustration above shows that only mobile user m_1, m_7 and m_9 satisfied this requirement, and subsequently registered. These will be recorded as a transaction t_n in each pass.

Next pass for the sliding window is to slide the window one step forward, and now the sliding window have a coverage from $t = 1$ to $t = 5$. This process is repeated until the sliding window covers from $t = 16$ to $t = 20$. For each pass, a set of mobile users who satisfied to be close at the same time for the time_threshold duration is registered. A list of them will be displayed here. We call them valid groups, as defined in the group pattern definition paper. Figure 2 illustrates.

Figure 3 shows the *support* counting for mobile users and its subsequent vertical representation of *support*. Support threshold $support_{\text{threshold}}$ is defined. Mobile user m

Pass 1 ($t = 1 \ldots 4$)	m_1, m_7, m_9
Pass 2 ($t = 2 \ldots 5$)	m_7, m_8
Pass 3 ($t = 3 \ldots 6$)	m_5, m_7, m_8
Pass 4 ($t = 4 \ldots 7$)	m_7, m_8
Pass 5 ($t = 5 \ldots 8$)	m_2, m_7
Pass 6 ($t = 6 \ldots 9$)	m_6, m_7, m_{10}
Pass 7 ($t = 7 \ldots 10$)	m_7
Pass 8 ($t = 8 \ldots 11$)	m_7
Pass 9 ($t = 9 \ldots 12$)	m_7
Pass 10 ($t = 10 \ldots 13$)	m_3, m_7
Pass 11 ($t = 11 \ldots 14$)	m_7
Pass 12 ($t = 12 \ldots 15$)	m_7, m_8
Pass 13 ($t = 13 \ldots 16$)	m_7, m_8, m_{10}
Pass 14 ($t = 14 \ldots 17$)	m_7, m_8, m_{10}
Pass 15 ($t = 15 \ldots 18$)	m_4, $\mathrm{m}_7, \mathrm{m}_8, \mathrm{m}_{10}$
Pass 16 ($t = 16 \ldots 19$)	m_4, $\mathrm{m}_7, \mathrm{m}_8, \mathrm{m}_{10}$
Pass 17 ($t = 17 \ldots 20$)	m_4, $\mathrm{m}_7, \mathrm{m}_8, \mathrm{m}_{10}$

Fig. 2. Records of transaction for all sliding window passes

Support for Mobile Users	*Vertical Representation of Support*
m_1: 1	m_1: 1
m_2: 1	m_2: 5
m_3: 1	m_3: 10
m_4: 3	m_4: 15, 16, 17
m_5: 1	m_5: 3
m_6: 1	m_6: 6
m_7: 17	m_7: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17
m_8: 9	m_8: 2, 3, 4, 12, 13, 14, 15, 16, 17
m_9: 1	m_9: 1
m_{10}: 6	m_{10}: 6, 13, 14, 15, 16, 17

Fig. 3. Calculating support for mobile users and vertical representation of support

is not considered if their *support* $m_{\text{support}} < support_{\text{threshold}}$. Let $support_{\text{threshold}}$ be 3, only m_4, m_7, m_8, and m_{10} will be considered. Algorithm now will proceed taking the supported mobile users to generate k-2 itemset from (m_4, m_7, m_8, and m_{10}). The subsequent combination for k-2 itemset are [(m_4, m_7), (m_4, m_8), (m_4, m_{10}), (m_7, m_8), (m_7, m_{10}) and (m_8, m_{10})].

Figure 4 illustrates the valid static group pattern mining (*SGPM*) process. The defined *support* = 3. It is now time to test the confidence of valid groups for high degree of *confidence*. Confidence is defined as $confidence_{\text{threshold}}$, and *confidence* for a particular combination of mobile user itemset such as (m_7, m_8, m_{10}) is defined as:

$$\frac{\sup port(m_7 \cap m_8 \cap m_{10})}{\sup port(m_7 \cup m_8 \cup m_{10})}$$

k-2 itemsets	*k-3 itemsets*	*k=4 itemsets*
$(m_4, m_7) : 3$	$(m_4, m_7, m_8) : 3$	$(m_4, m_7, m_8, m_{10}) : 3$
$(m_4, m_8) : 3$	$(m_7, m_8, m_{10}) : 5$	
$(m_4, m_{10}) : 3$	$(m_4, m_8, m_{10}) : 3$	
$(m_7, m_8) : 9$	$(m_4, m_7, m_{10}) : 3$	
$(m_7, m_{10}) : 4$		
$(m_8, m_{10}) : 5$		
Support = 3	*Support* = 3	*Support* = 3
∴ Select All	∴ Select All	∴ Select (m_4, m_7, m_8, m_{10})

Fig. 4. Valid static group pattern mining process demonstration

Support for maximal itemset $(m_4 \cap m_7 \cap m_8 \cap m_{10})$ is 3. Support for $(m_4 \cup m_7 \cup m_8 \cup m_{10})$ is 17. The confidence of valid group pattern (m_4, m_7, m_8, m_{10}) is *17%*. Confidence is used to confirm that within the whole time horizon for that *area* a_1, from $t = 0$ to $t = 20$, altogether 17 records generated from the sliding window, the ratio of (m_4, m_7, m_8, m_{10}) is present within the same transaction, compared to transactions containing either one of m_4, m_7, m_8 or m_{10}. Confidence is therefore, subject to the size of *time horizon*, and the *frequency* of occurrence of individual item. In the interest of space, we do not show how this problem is dealt in this paper.

Algorithm *Sliding-Window*
Input: User movement database grouped by area a_n, variable $t_{threshold}$
Output: m_{record} of mobile users who is present in the whole sliding window

```
result = ∅;
S_width = t_threshold; // defining width of sliding window
for (S_ref = 0; (S_ref + t_threshold != t_horizon); t ++) do begin
    for (m_i = 1; m_i ≤ m_j; m ++) do begin
        for (m_i.start; m_i.start < m_i.finish; m_i ++) do begin
            if (m_i.t_ref == ∅) skip;
            append(result, m_i);
        end for
    end for
end for
return result;
```

Fig. 5. Algorithm *Sliding-Window*

Figure 5 represents algorithm *Sliding-Window* where the sliding window is defined by $t_{threshold}$, and the program code for how the sliding window slides through the database. In order for a mobile user to be recorded, a mobile user must be within the sliding window, be present at all times from sliding window t_{ref} to $t_{ref} + t_{threshold}$. If the mobile user is not present, it will not be recorded. If the mobile user is present at all times, it will be recorded for *AGSP* algorithm.

Algorithm AGSP
Input: *result* from algorithm *Sliding-Window*, $support_{threshold}$
Output: List supported itemsets

```
R1 = {large r-itemsets} // R gathered from result
for (k=2; Rk-1 ≠ ∅; k ++) do begin
    Rk = apriori-gen(Rk-1);
    for all transactions t ε R do begin
        Rt = subset (Rk, t)
        for all candidates r ε Rt do begin
            r.count++;
        Rk = {r ε Rk | r.count ≥ support_threshold}
return Rk;
```

Fig. 6. Algorithm *AGSP*

Figure 6 shows the algorithm *AGSP* where the result from *Sliding-Window* algorithm is given in order to generate a list of frequent combinations of itemsets similar to *Apriori* algorithm. For instance, only mobile users who have $support \geq support_{threshold}$ is considered for combination generation. The process is repeated until no further combinations can be generated, and the resulting output is a combination of mobile users (m_i, …, m_j) where they are highly supported from the result generated from *Sliding-Window* algorithm.

Resulting output is a combination of valid group pattern, where (m_i, …, m_j) is located within the same area, near to each other, for a good duration $t_{threshold}$. This shows evidence of them being close together frequently enough within the same area and time for at least $t_{threshold}$. In order to find out the ratio of time that this combination (m_i, …, m_j) over the total duration of records R from *sliding-window*, apply the formula of $confidence = (m_i, \cap \ldots \cap, m_j) / (m_i, \cup \ldots \cup, m_j)$.

5 Evaluation

In this section, we evaluate and compare the performance between *ASGP* and *AGP* algorithms. The experiments has been conducted using synthetically generated user movement database on a Pentium IV machine with a CPU clock rate of 2.8 Ghz, and 504 MB of main memory. Note that both dataset and program are executed in main memory so that it represents execution time without bottlenecks from disk access. We compare the time it requires *AGP* algorithm and *ASGP* algorithm to access from user movement database, perform mining and generating the result of all the valid group patterns.

5.1 Dataset

Since real dataset are not available, we have implemented a synthetic user movement database generator for our experiment. Figure 7 shows the parameters used in performance evaluation for dataset *T5.I2.D1000*, *T10.I2.D1000*, *T5.I4.D1000*, *T10.I4.D1000*. Fig 8 represents the input parameters, where *D* represents the number of records, *T* represents the average size of record, and *I* represent the average size of maximal potentially large item sets.

Dataset	*D*	*T*	*I*	*Size (MB)*
T5.I2.D1000	1000	5	2	9.76
T10.I2.D1000	1000	10	2	19.53
T5.I4.D1000	1000	5	4	18.25
T10.I4.D1000	1000	10	4	39.06

Fig. 7. Dataset parameters for performance evaluation

5.2 Results

Figure 8 illustrates the evaluation results for *T5.I2.D1000, T10.I2.D1000, T5.I4.D1000, and T10.I4.D1000* respectively. It can be observed that on all occasions, algorithm *ASGP* takes shorter time to generate valid group patterns. When the *support* threshold is set to very high (i.e. 1.0) both algorithm takes roughly the same time to generate result, because there are very limited amount of candidates in the dataset for traversal. As the *support* is reduced from 1.0 to 0.1 through each decrement of 0.1, the number of potential candidates becomes larger and larger. *ASGP* takes a shorter time than *AGP* generally from *support* = 0.9 to *support* = 0.3, and after this both algorithms takes roughly the same time to generate valid group patterns. This is because *support* is low, and there are many potential candidates, and more processing time required. Nevertheless, algorithm *ASGP* still outperforms algorithm *AGP* at a varying degree, from slightly quicker for very large dataset to much quicker for a moderate sized dataset.

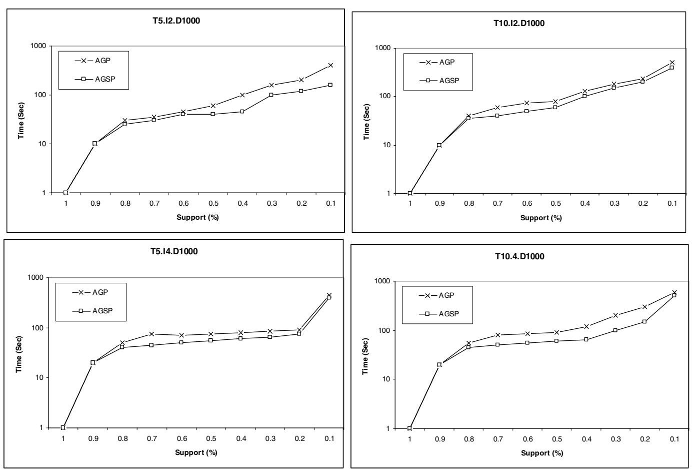

Fig. 8. Execution time required between ASGP and AGP algorithms

6 Conclusion

This paper reports an innovative redefinition of group pattern mining, called Static Group Pattern Mining (*SGPM*). The objective of this research work is to address the bottlenecks of *AGP* algorithm. Instead of using Eucilidean distance and calculate the distance for each and every pair of mobile users over the time horizon, *SGPM* uses the concept of area, sliding window and Apriori-like algorithm to find all valid groups, and valid group patterns. Performance evaluations have shown that *SGPM* have quicker execution time than *AGP* algorithm in 4 out of 4 cases. Future work from here is to further improve the execution time of valid group pattern mining problem and addressing obstacle issues in the mobile environment.

References

1. R. Agrawal and R. Srikat. Fast Algorithms for Mining Association Rules. In Proc. of the 20th VLDB, 1994.
2. R. Agrawal and R. Srikat. Mining Sequential Patterns. In Proc. of 11th ICDE, 1995.
3. B. Hofmann-Wellenhof, H. Lichtenegger, and J. Collins. Global Positioning System: Theory and Practice. Springer-Verlag Wien New York, third revised edition, 1994.
4. S. Chakrabarti, S. Sarawagi, and B. Dom. Mining Surprising Patterns using Temporal Description Length. In Proc. of 24th VLDB, 1998.
5. L. Forlizzi, R. H. Guting, E. Nardelli, and M. Schneider. A Data Model and Data Structures for Moving Objects Databases. ACM SIGMOD Record, 2000.
6. D. R. Forsyth. Group Dynamics. Wadsworth, Belmont, CA, 1999.
7. J. Han, G. Dong, and Y. Yin. Efficient Mining of Partial Periodic Patterns in Time Series Database. In Proc. of 15th ICDE, 1999.
8. J. Han, J. Pei, and Y. Yin. Mining Frequent Patterns Without Candidate Generation. In Proc. of ACM SIGMOD, 2000.
9. J. Han and A. W. Plank. Background for Association Rules and Cost Estimate of Selected Mining Algorithms, In Proc. of the 5th CIKM, 1996.
10. K. Koperski and J. Han. Discovery of Spatial Association Rules in Geographical Information Databases. In Proc of 4th Int Symp. on Advances in Spatial Databases, 1995.
11. J. F. Roddick and B. G. Lees. Paradigms for Spatial and Spatio-Temporal Data Mining. Geographic Data Mining and Knowledge Discovery, 2001.
12. J. F. Roddick and M. Spiliopoulou. A Survey of Temporal Knowledge Discovery Paradigms and Methods. IEEE Trans. on Knowledge and Data Engineering, 2002.
13. Wei Wang, Jiong Yang, and P. S. Yu. InfoMiner+: Mining Partial Periodic Patterns in Time Series Data. IEEE Transaction on Knowledge and Data Engineering, 2002.
14. Paul Zarchan. Global Positioning System: Theory and Applications, vol I. American Institute of Aeronautics and Astronautics, 1996.
15. Reed Electronics Research RER – The mobile phone industry – a strategic overview, October 2002.
16. U. Varshney, R. Vetter, and R. Kalakota. Mobile commerce: A new frontier. IEEE Computer: Special Issue on E-commerce, pages 32-38, October 2000.
17. Yida Wang, Ee-Peng Lim, and San-Yih Hwang. On Mining Group Patterns from Mobile Users. In Proc. of the 14th International Conference on Database and Expert Systems Applications – DEXA 2003, Prague, Czech Republic, 1-5 Sep, Lecture Notes in Computer Science vol. 2736, pp. 287-296, 2003.
18. Yida Wang, Ee-Peng Lim, and San-Yih Hwang. Efficient Group Pattern Mining Using Data Summarization. In Proc. of the 15th International Conference on Database and Expert Systems Applications – DEXA 2004, Beijing, China, Lecture Notes in Computer Science vol. 2973, pp. 895-907, 2004.