On Mining Group Patterns of Mobile Users

Yida Wang[1], Ee-Peng Lim[1], and San-Yih Hwang[2]

[1] Centre for Advanced Information Systems,
School of Computer Engineering
Nanyang Technological University,
Singapore 639798, Singapore
wyd66@pmail.ntu.edu.sg
aseplim@ntu.edu.sg
[2] Department of Information Management
National Sun Yat-Sen University,
Kaohsiung, Taiwan 80424
syhwang@mis.nsysu.edu.tw

Abstract. In this paper, we present a *group pattern mining* approach to derive the grouping information of mobile device users based on the spatio-temporal distances among them. Group patterns of users are determined by a distance threshold and a minimum duration. To discover group patterns, we propose the AGP and VG-growth algorithms that are derived from the Apriori and FP-growth algorithms respectively. We further evaluate the efficiencies of these two algorithms using synthetically generated user movement data.

1 Introduction

In our daily activities, we are affiliated to groups of many different sorts. Group dynamics and its influence on individual decision making have been well studied by sociologists[6], and it has been shown that peer pressure and group conformity can affect the buying behaviors of individuals. With a good knowledge of groups a customer belongs to, one can derive common buying interests among customers, develop group-specific pricing models and marketing strategies, and provide personalized services. For example, venders may offer discounts or recommend products to groups so as to encourage more purchases with a higher success rate.

There are many ways one can determine the groups a person belongs to, e.g., partitioning people into groups based on past purchases of same product items, similar occupations, incomes, etc.. In this paper, we aim to derive group knowledge of users using their spatio-temporal information, namely their movement data. The grouping knowledge derived from these movement data is unique compared to the previous approaches in several ways:

– *Physical proximity between group members*: The group members are expected to be physically close to one another when they acts as a group. Such characteristics are common among many types of groups, e.g., shopping pals, game partners, etc..

V. Mařík et al. (Eds.): DEXA 2003, LNCS 2736, pp. 287–296, 2003.

– *Temporal proximity between group members*: The group members are expected to stay together for some meaningful duration when they acts as a group. Such characteristic distinguishes an ad hoc cluster of people who are physically close from a group of people who come together for some planned activity(ies).

Unlike the existing techniques that partition people into groups based on other factors, the above two characteristics ensure that members of the derived groups are aware of and maintain contact with one another. Hence, the group members are expected to exert much stronger influence on one another.

Related Work

In this research, we assume that the user movement data can be collected by logging location data emitted from mobile devices. This logging facility provides time series of locations for each user. This assumption is technically feasible since mobile devices are becoming more and more location-aware using positioning technologies [3,15], which has become more affordable and even more so in the future. To keep a focused discussion, we shall keep the privacy and legal issues out the scope of this paper.

Group pattern mining deals with time series of user location information involving temporal and spatial dimensions. We observe that previous temporal and spatial data mining research mostly focus either on temporal or spatial mining[4, 10,12], not both. There are also significant work in periodicity analysis for time series data [7,13,14]. Nevertheless, these time series data usually do not involve spatial information. Although there has been some work on spatial-temporal mining [11,5] that considers both temporal and spatial aspects of information, they mainly focus on the models and structures for indexing the moving objects.

2 Problem Definition

The *user movement database*, D, consists of a set of time series of locations, one for each user. Assume that there are M distinct users $u_1, u_2, \cdots, u_M$. D is defined as the union of time series of locations belonging to all the users, i.e., $D = \cup_{i=1}^{M} D_i$. Each D_i is a time series containing triplets (t, x, y) denoting the x- and y-coordinates respectively of user u_i at time t. For simplicity, we assume that the all user locations are known at every time point, and the interval between every t and $t + 1$ is fixed.

Definition 1. *Given a group of users G, a maximum distance threshold max_dis, and a minimal time duration threshold min_dur, a set of consecutive time points $[t, t + k]$ is called a* ***valid segment*** *of G if*

1. *All users in G are not more than max_dis apart at time t, $t + 1$, $\cdots$, and $t + k$;*
2. *Some users in G are more than max_dis apart at time $t - 1$;*

Table 1. User Movement Database D

u_1			u_2			u_3			u_4			u_5			u_6		
t	x	y	t	x	y	t	x	y	t	x	y	t	x	y	t	x	y
0	68	41	0	73	41	0	73	46	0	81	39	0	80	43	0	99	43
1	72	75	1	72	69	1	79	71	1	71	67	1	71	71	1	61	97
2	79	51	2	80	52	2	82	59	2	81	53	2	73	51	2	34	45
3	80	50	3	84	52	3	81	53	3	85	57	3	80	11	3	42	96
4	62	56	4	59	10	4	50	63	4	60	53	4	58	9	4	7	80
5	45	65	5	24	49	5	49	61	5	22	45	5	20	48	5	29	54
6	67	58	6	39	19	6	36	27	6	40	19	6	40	19	6	39	61
7	73	53	7	68	52	7	72	52	7	74	53	7	72	53	7	88	35
8	75	51	8	72	51	8	69	54	8	73	53	8	75	53	8	62	70
9	73	53	9	64	56	9	62	50	9	74	51	9	79	53	9	7	59

3. Some users in G are more than max_dis apart at time $t + (k + 1)$;
4. $(k + 1) \geq min\_dur$;

Consider the user movement database in Table 1. For $min\_dur = 3$ and $max\_dis = 10$, $[5, 8]$ is a valid segment of the user group $\{u_2, u_4\}$.

Definition 2. *Given 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.*

The valid segments of the group pattern P are therefore the valid segments of its G component. We also call a group pattern with k users a **k-group pattern**.

The thresholds max_dis and min_dur are used to define the spatial and temporal proximity requirements between members of a group. By choosing appropriate thresholds, we can define the minimum duration a set of users must "stay close together" before we consider them as a meaningful group.

In a user movement database, a group pattern 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 $s_1, \cdots, s_n$, and N denotes the number of time points in the database, the **weight** of P is defined as:*

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

Since weight represents the *proportion* of the time points when the group of users stay close, the larger the weight, the more significant is the group pattern. If the weight of a group pattern exceeds a threshold min_wei, we call it a **valid group pattern**, and the corresponding group of users a **valid group**. For example, 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 weight $6/10 \geq 0.5$.

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

3 AGP: Algorithm Based on Apriori Property

In this section, we present the AGP (**A**priori-like algorithm for mining valid **G**roup **P**atterns) algorithm, which is derived from the well known Apriori algorithm[1] as the Apriori property also holds for group patterns.

Definition 5. *Given two group patterns,* $P =< G, max\_dis, min\_dur >$ *and* $P' =< G', max\_dis, min\_dur >$, P' *is called a* ***sub-group pattern*** *of* P *if* $G' \subseteq G$.

Property 1. [Apriori property for group patterns]: Given database D and thresholds max_dis, min_dur, and min_wei, if a group pattern is *valid*, all of its *sub-group patterns* will also be valid.

This property can be proven quite easily and we shall leave out the proof in the interest of space.

Based on the Apriori property, we develop the AGP algorithm as shown in Figure 1[1]. In the algorithm, we use C_k to denote the set of candidate k-groups, and use $\mathbb{G}_k$ to denote the set of valid k-groups. From $\mathbb{G}_1$, the set of all distinct users, the algorithm first computes $\mathbb{G}_2$, which is in turn used to compute $\mathbb{G}_3$. The process repeats until no more valid k-groups can be found. In each iteration, AGP performs *join* operation to generate candidate k groups from $\mathbb{G}_{k-1}$, and the generated candidates are further pruned using Apriori property.

```
Input:  D, max_dis, min_dur, and min_wei
Output: all valid groups 𝔾
01  𝔾_1 = all distinct users;
02  for (k = 2; 𝔾_{k-1} ≠ ∅; k++)
03      C_k = Generate_Candidate_Groups(𝔾_{k-1});
04      for (t = 0; t < |D|; t++)    // scan D to compute the "weight"
05          for each candidate k-group c_k ∈ C_k
06              if Is_Close(c_k, t, max_dis) then   // check closeness of candidate group c_k
07                  c_k.cur_seg++;
08              else
09                  if c_k.cur_seg ≥ min_dur then
10                      c_k.weight = c_k.weight + c_k.cur_seg;
11                  c_k.cur_seg = 0;
12      𝔾_k = {c_k ∈ C_k | c_k.weight ⩾ min_wei × N};
13      𝔾 = 𝔾 ∪ 𝔾_k;
14  return 𝔾;
```

Fig. 1. Apriori-like Algorithm AGP for Mining Group Patterns.

Let M be the number of distinct users and N be the number of time points in D. The time complexity of AGP algorithm is $O(\ \Sigma_k\{k \cdot |\mathbb{G}_{k-1}|^3,\ M \cdot N,\ N \cdot |C_k| \cdot k^2\ \}\ )$ (please refer to [9] for a detailed discussion on the time complexity of Apriori-like algorithm).

[1] Some functions are not shown to save space.

4 VG-Growth: An Algorithm Based on Valid Group Graph Data Structures

AGP, similar to the Apriori algorithm, incurs large overheads in candidate k-group pattern generation and database scans to check if the candidates are valid. In order to reduce such overheads, we propose a divide-and-conquer algorithm *VG-growth* using a novel data structure known as *VG-graph*. VG-growth and VG-graph are designed based on the principle similar to that of FP-growth and FP-tree (Frequent Pattern tree) for association rule mining[8].

Definition 6. *A **valid group graph** (or **VG-graph**) is a directed graph* (V, E)*, where* V *is a set of vertices representing users in the set of valid 2-groups, and* E *is a set of edges representing the set of valid 2-groups. Each edge is also associated with the valid segments of the corresponding valid 2-group pattern.*

To construct a VG-graph, a complete scan on D is required to compute the valid 2-group patterns using the AGP algorithm. The users represented by V in the VG-graph are called the **valid users**. For easy enumeration of all the edges in a VG-graph, the edge linking two users in a valid 2-group pattern always origins from the user with a smaller id. Consider the movement database D in Table 1. Given $max\_dis = 10$, $min\_dur = 3$ and $min\_wei = 60\%$, we construct the corresponding VG-graph based on the set of valid 2-groups associated with valid segments, as shown in Figure 2.

Valid 2-groups	Valid segment lists
$\{u_1, u_2\}$	$s(u_1,u_2)=\{[0,3],[7,9]\}$
$\{u_1, u_4\}$	$s(u_1,u_4)=\{[1,4],[7,9]\}$
$\{u_2, u_3\}$	$s(u_2,u_3)=\{[0,3],[6,9]\}$
$\{u_2, u_4\}$	$s(u_2,u_4)=\{[0,3],[5,8]\}$
$\{u_2, u_5\}$	$s(u_2,u_5)=\{[0,2],[4,8]\}$
$\{u_3, u_4\}$	$s(u_3,u_4)=\{[1,3],[6,8]\}$
$\{u_4, u_5\}$	$s(u_4,u_5)=\{[0,2],[5,9]\}$

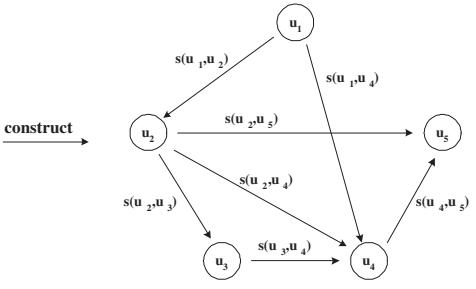

Fig. 2. The VG-graph for Table 1

Note that VG-graph only contains the valid users involved in valid 2-groups. Thus, the number of vertices in the VG-graph will be no more than the total number of users. Suppose that on average, each user belong to k valid 2-groups. It can be shown that VG-graph will take smaller space than the original database if each user participates in less than $2 \times min\_dur$ valid 2-groups.

Based on the VG-graph data structure, we develop the VG-growth algorithm presented as follows.

If $(u \rightarrow v)$ is a directed edge in a VG-graph, u is called the **prefix-neighbor** of v. For example, in Figure 2, u_2, u_4 are the prefix-neighbors of u_5.

Consider the VG-graph in Figure 2. We can mine the valid group patterns by traversing the VG-graph. In the following, we illustrate how u_4 in the VG-graph can be traversed.

Input: VG-graph, *max_dis*, *min_dur*, and *min_wei*
Output: all valid groups
Method: call procedure *VG-growth(VG-graph, null).*
Procedure: *VG-growth(Graph, α)*

```
01  for each vertex u in Graph
02      generate the condition group β = {u} ∪ α;
03      select all prefix-neighbors of u, denoted by V_β;
04      if V_β ≠ ∅ then
05          for each vertex v in V_β
06              output a valid group: {v} ∪ β;
07          select the directed edges on V_β, denoted by E(V_β);
08      if E(V_β) ≠ ∅ then
09          for each directed edge (v_i → v_j) in E(V_β)
10              s(v_i v_j) = s(v_i v_j) ∩ s(v_i u) ∩ s(v_j u);   // adjust against u
11              if s(v_i v_j) does not satisfy min_dur and min_wei then
12                  remove edge (v_i → v_j) from E(V_β);
13      if E(V_β) ≠ ∅ then
14          construct the conditional VG-graph of β, VG(β);
15          call procedure VG-growth(VG(β), β);
```

Fig. 3. VG-growth Algorithm.

– Select all prefix-neighbors of u_4, $V_{u_4} = \{u_1, u_2, u_3\}$. Three valid 2-groups, $\{u_1, u_4\}$, $\{u_2, u_4\}$, $\{u_3, u_4\}$, are generated. Next, select the directed edges on V_{u_4}, $E(V_{u_4}) = \{(u_1 \rightarrow u_2), (u_2 \rightarrow u_3)\}$ with associated segment lists, $s(u_1, u_2) = \{[0, 3], [7, 9]\}$, $s(u_2, u_3) = \{[0, 3], [6, 9]\}$. Adjust the two segment lists against u_4 as follows.
  - $s(u_1, u_2) = s(u_1, u_2) \cap s(u_1, u_4) \cap s(u_2, u_4) = \{[0, 3], [7, 9]\} \cap \{[1, 4], [7, 9]\} \cap \{[0, 3], [5, 8]\} = \{[1, 3], [7, 8]\}$
  - $s(u_2, u_3) = s(u_2, u_3) \cap s(u_2, u_4) \cap s(u_3, u_4) = \{[0, 3], [6, 9]\} \cap \{[0, 3], [5, 8]\} \cap \{[1, 3], [6, 8]\} = \{[1, 3], [6, 8]\}$

  Check these adjusted segment lists against *min_dur* and *min_wei*, and remove those edges that does not meet the threshold requirements. The edge $(u_1 \rightarrow u_2)$ is therefore removed.
  The prefix-neighbors V_{u_4} and $E(V_{u_4})$ (after adjustment and checking) form u_4's **conditional group base**. From the u_4's conditional group base (involving u_1, u_2, and u_3 and an edge between u_2 and u_3), we derive the the **conditional VG-graph** of u_4, denoted by $VG(u_4)$, which contains two vertices $\{u_2, u_3\}$ and an edge $(u_2 \rightarrow u_3)$ with associated segment list $\{[1, 3], [6, 8]\}$.
  We perform mining recursively on $VG(u_4)$ and any valid groups generated from $VG(u_4)$ will involve u_4 as a member. We compute $V_{u_3 u_4} = \{u_2\}$ and the valid 3-groups $\{u_2, u_3, u_4\}$ is generated. Since u_2 has no prefix-neighbors, the mining process for u_4 terminates.

Given a vertex u, let V_u denote the set of prefix-neighbors of u, and $E(V_u)$ be the set of directed edges after adjustment against u. V_u and $E(V_u)$ form a small database of groups which co-occur with u in some valid groups, known as u's **conditional group base**. We can compute all the valid groups associated with u in u's conditional group base by creating a smaller VG-graph, known as

u's **conditional VG-graph** and denoted by $VG(u)$. The mining process can be recursively performed on the conditional VG-graph. The complete VG-growth algorithm is given in Figure 3.

5 Evaluation

In this section, we evaluate and compare the performance of AGP and VG-growth algorithms based on their execution time. The experiments have been conducted using synthetically generated user movement databases on a Pentium-IV machine with a CPU clock rate of 2.4 GHz, and 1 GB of main memory. Note that both AGP and VG-growth are implemented to run in main memory to give a direct comparison between them.

5.1 Methodology

Performance Study. In the performance study experiment, we measure the execution times of AGP and VG-growth on three synthetic datasets (see Table 3) for different *min_wei* thresholds (from 0.1% to 10%). The thresholds *max_dis* and *min_dur* are fixed as 50 and 4 respectively. The execution time for VG-growth includes the time for constructing VG-graph.

Scale-up Performance. In scale-up experiment, we study the scale-up features of VG-growth against both the number of users (M) and the number of time points in the database (N). We use different M values from 1000 to 5000 with N fixed as 1000. We also vary N from 1000 to 10,000 keeping M fixed as 1000. The *min_wei* threshold varies from 0.5% to 10%. The thresholds *max_dis* and *min_dur* are fixed as 50 and 4 respectively. The scale-up feature of AGP is not included as in the performance study we have found that VG-growth always outperforms AGP.

5.2 Datasets

Since real datasets are not available, we have implemented a synthetic user movement database generator for our experiments. Our data generation method extends that for transaction databases described in [1,2]. The process of generation can be divided into 2 steps:

1. Generate a set $\mathbb{G}$ of maximal valid groups[2];
2. Pick groups from $\mathbb{G}$ and "assign" them to each time point by giving them locations that are close at the time point.

Due to space constraint, we shall not elaborate the detailed steps of database generation. Table 2 shows the parameters used for synthetic database generation. Table 3 summarizes the parameter settings for performance study and scale-up performance.

[2] Given a set $\mathbb{G}$ of group patterns and a group pattern $P \in \mathbb{G}$, we call P a **maximal group pattern**, and the group in P a **maximal group**, if P is not a sub-group pattern of any other group pattern in $\mathbb{G}$.

Table 2. Parameter List

M	The number of distinct users
$N_{\mathbb{G}}$	The number of potentially maximal valid groups in the set $\mathbb{G}$
A_G	The average size of the potentially maximal valid groups
N	The number of time points(i.e., the whole time span)
A_t	The average number of groups involved in each time point
A_d	The average time duration of each group

Table 3. Parameter Settings

Performance Study & Scale-up against N							
Name	M	A_G	$N_{\mathbb{G}}$	N	A_d	A_t	Size in Megabytes
DBI	1,000	5	1,000	1,000	6	100	15.4
DBII	1,000	5	1,000	5,000	6	100	81.5
DBIII	1,000	5	1,000	1,0000	6	100	164
Scale-up against M							
Name	M	A_G	$N_{\mathbb{G}}$	N	A_d	A_t	Size in Megabytes
DBIV	1,000	5	1,000	1,000	6	100	15.4
DBV	3,000	5	3,000	1,000	6	300	46.3
DBVI	5,000	5	5,000	1,000	6	500	77.3

5.3 Results

Performance Study. The results of performance study are shown in Figure 4, in which the Y-axis has a log scale. It is observed that VG-growth outperforms AGP for all the datasets, especially when *min_wei* becomes smaller ($< 1\%$). When *min_wei* is small, there will be more valid groups as shown in Figure 5. The cost of candidate groups generation in AGP will become very high due to multiple database scans to check the closeness of members in the candidate groups.

We also observe that the time to find valid 2-groups is significant compared to the total execution time. In particular, for large *min_wei* values, VG-growth spends almost all the time finding valid 2-groups as shown in Figures 4 and 5. When the proportion of valid 2-groups is large, VG-growth takes almost the same time as AGP. This does not come as a surprise since VG-growth uses the same method as AGP to find valid 2-groups. Hence, reducing the cost of finding valid 2-groups is an important topic for our future work. Nevertheless, considering that the same VG-graph can be used for different runs of group pattern mining, the effective execution time of VG-growth is actually much less if we amortise the construction time of VG-graph over the runs.

In the performance study, we also examine the size of VG-graphs. Assuming an adjacency list structure, we compute the estimated size of VG-graph. Figures 6 and 7 give the sizes of VG-graphs in KB for different *min_wei*, and the compression ratios respectively. Although the actual sizes of VG-graphs are different for each *min_wei*, the compression ratios are almost the same and the compression ratios range from $0.5\% \sim 5\%$. This shows the compactness of the VG-graphs.

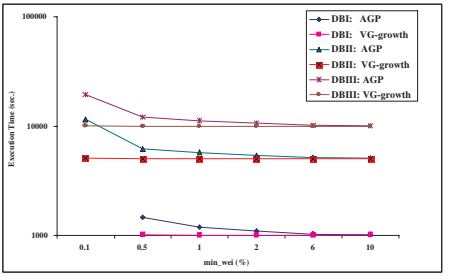

Fig. 4. Performance Study.

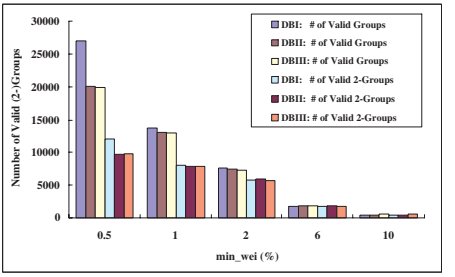

Fig. 5. Number of Valid (2-)Groups.

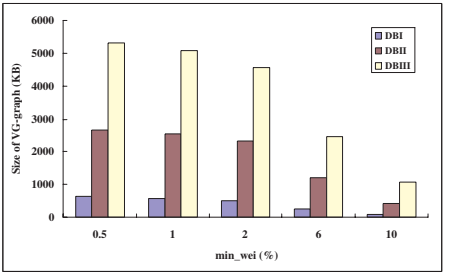

Fig. 6. Size of VG-graph.

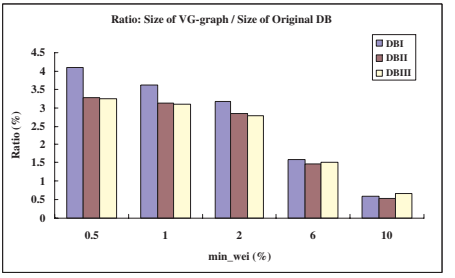

Fig. 7. Compression Ratio of VG-graph.

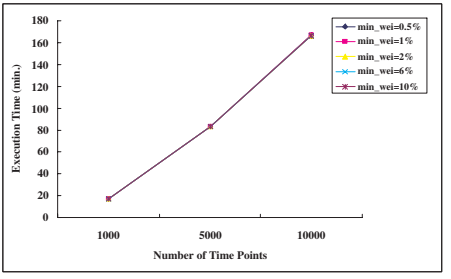

Fig. 8. Scale-up with N (VG-growth).

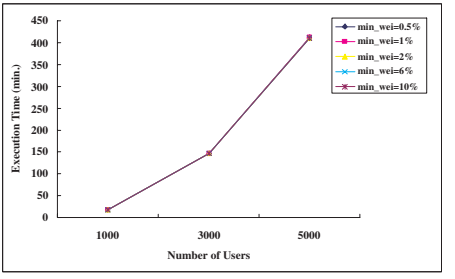

Fig. 9. Scale-up with M (VG-growth).

Scale-up Performance. The results of scale-up performance experiments for VG-growth are shown in Figures 8 and 9. The performance curves for different *min_wei* in each figure are almost identical. The execution time is almost linear with the number of time points, N, and is almost linear with M^2, where M is the number of distinct users. This due to the fact that (1) the time required for finding valid 2-groups occupies a large proportion of the total execution time; and (2) most of the time required to find valid 2-groups is spent on scanning the database to compute the weights of candidate 2-groups, which is roughly determined by $N \cdot \binom{M}{2}$.

6 Conclusions

This paper reports a innovative approach to mine user group patterns from their movement data. The discovered group patterns, satisfying both spatial and temporal proximity requirements, could potentially be used in target marketing and personalized services. We formally define the notion of group patterns and develop two algorithms (AGP and VG-growth) for mining valid group patterns. The performance of these two algorithms has been reported using synthetically generated user movement databases. It has been shown that the cost of mining group patterns is mainly due to the computation of valid-2 group patterns as the number of larger group patterns reduces. Hence, the performance gain of VG-growth algorithm is most apparent when the *min_wei* is small. However, considering that VG-graphs can be stored beforehand and its construction cost can be amortised over multiple runs of group pattern mining, the savings of VG-growth algorithm will be more significant.

References

1. R. Agrawal and R. Srikant. Fast Algorithms for Mining Association Rules. In *Proc. of the 20th VLDB*, 1994.
2. R. Agrawal and R. Srikant. 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 Geographic 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 with Gap Penalties. In *Proc. of the 2nd ICDM*, 2002.
14. Jiong Yang, Wei Wang, and Philip Yu. Mining Asynchronous Periodic Patterns in Time Series Data. *IEEE Transaction on Knowledge and Data Engineering*, 2002.
15. Paul Zarchan. *Global Positioning System: Theory and Applications*, volume I. American Institute of Aeronautics and Astronautics, 1996.