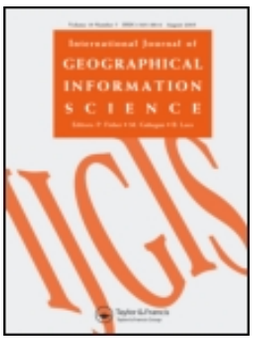

International Journal of Geographical Information Science

ISSN: 1365-8816 (Print) 1362-3087 (Online) Journal homepage: https://www.tandfonline.com/loi/tgis20

Visual mining of moving flock patterns in large spatio-temporal data sets using a frequent pattern approach

Ulanbek Turdukulov, Andres Oswaldo Calderon Romero, Otto Huisman & Vasilios Retsios

To cite this article: Ulanbek Turdukulov, Andres Oswaldo Calderon Romero, Otto Huisman & Vasilios Retsios (2014) Visual mining of moving flock patterns in large spatio-temporal data sets using a frequent pattern approach, International Journal of Geographical Information Science, 28:10, 2013-2029, DOI: 10.1080/13658816.2014.889834

To link to this article: https://doi.org/10.1080/13658816.2014.889834

Published online: 14 Mar 2014.

International Journal of Geographical Information Science, 2014
Vol. 28, No. 10, 2013–2029, http://dx.doi.org/10.1080/13658816.2014.889834

Taylor & Francis
Taylor & Francis Group

Visual mining of moving flock patterns in large spatio-temporal data sets using a frequent pattern approach

Ulanbek Turdukulov[a]*, Andres Oswaldo Calderon Romero[b], Otto Huisman[c] and Vasilios Retsios[a]

[a]*Faculty of Geo-Information Science and Earth Observation (ITC), University of Twente, Enschede, Overijssel, The Netherlands;* [b]*Grupo de Investigacion Aplicado en Sistemas (GRiAS), Universidad de Nariño, Pasto, Colombia;* [c]*ROSEN Integrity Solutions, Lingen (Ems), Lower Saxony, Germany*

(*Received 27 June 2013; accepted 22 January 2014*)

The popularity of tracking devices continues to contribute to increasing volumes of spatio-temporal data about moving objects. Current approaches in analysing these data are unable to capture collective behaviour and correlations among moving objects. An example of these types of patterns is moving flocks. This article develops an improved algorithm for mining such patterns following a frequent pattern discovery approach, a well-known task in traditional data mining. It uses transaction-based data representation of trajectories to generate a database that facilitates the application of scalable and efficient frequent pattern mining algorithms. Results were compared with an existing method (Basic Flock Evaluation or BFE) and are demonstrated for both synthetic and real data sets with a large number of trajectories. The results illustrate a significant performance increase. Furthermore, the improved algorithm has been embedded into a visual environment that allows manipulation of input parameters and interactive recomputation of the resulting flocks. To illustrate the visual environment a data set containing 30 years of tropical cyclone tracks with 6 hourly observations is used. The example illustrates how the visual environment facilitates exploration and verification of flocks by changing the input parameters and instantly showing the spatio-temporal distribution of the resulting flocks in the Space-Time Cube and interactively selecting, querying and saving the resulting flocks for further analysis and verification.

Keywords: flock patterns; frequent pattern mining; spatio-temporal data sets; visual mining; Space-Time Cube; tropical cyclones

1. Introduction

Modern data acquisition techniques such as Global Positioning System (GPS), radio-frequency identification (RFID) and mobile phones have resulted in the collection of massive amounts of movement data in recent years. Increasing popularity of these technologies and the ubiquity of mobile devices imply that this volume of spatio-temporal data being collected will increase at accelerated rates in the future.

Due to the increasing availability of spatial-temporal databases, many different methodologies have been explored in the search for meaningful information hidden in this kind

*Corresponding author. Email: ulanbek.turdukulov@curtin.edu.au
Present address for Ulanbek Turdukulov: Department of Spatial Sciences, Western Australian School of Mines, Curtin University, Perth, Western Australia.

of data. Understanding of how diverse entities move in a spatial context has demonstrated their usefulness in topics as diverse as sports (Iwase and Saito 2002), socio-economic geography (Frank *et al.* 2001), animal migration (Dettki *et al.* 2004) and security and surveillance (Makris and Ellis 2002, Piciarelli *et al.* 2006).

Early approaches to discovering patterns from spatio-temporal data sets include ad hoc queries aimed at answering single predicate range or nearest neighbour queries. As a result, it is difficult to capture collective behaviour and correlations among the entities involved using this type of approach. Recently, a new demand for querying patterns capturing 'group' or 'common' behaviour among moving entities has emerged. Of particular interest has been the development of approaches that can identify groups of moving objects whose members share a strong relationship by being present within a defined spatial region during a given time duration. Some examples of these kinds of approaches are moving cluster analyses (Kalnis *et al.* 2005, Jensen *et al.* 2007), convoy queries (Jeung *et al.* 2008) and flock patterns (Gudmundsson and van Kreveld 2006, Benkert *et al*. 2008, Vieira *et al.* 2009).

Vieira *et al.* (2009) define moving flock patterns as groups of at least μ entities moving in the same direction while being close to one another during a given time interval δ (Figure 1). They consider group of trajectories to be close together if there exists a disk with a given radius ε that encloses all of them. The current approach to discover moving flock patterns consists of finding a set of such disks in each time instance and then merging the results from one time instance to another. As consequence, the performance and the number of final patterns depend on both the number of disks and how they are combined.

In parallel with developments in the fields of spatial databases and computational geometry, some areas of traditional data mining have also focused on discovering frequent patterns in general attribute data. Association rule learning and frequent pattern mining (FPM) (Han and Pei 2000) are popular and well-researched methods for discovering interesting relations between variables in large databases. These approaches were initially developed to solve a specific task in the commerce sector. Frequent patterns are itemsets, subsequences or substructures that can be said to be present in a data set if their frequency exceeds a user-specified threshold (Han and Kamber 2006). It is thought that approaches to find frequent

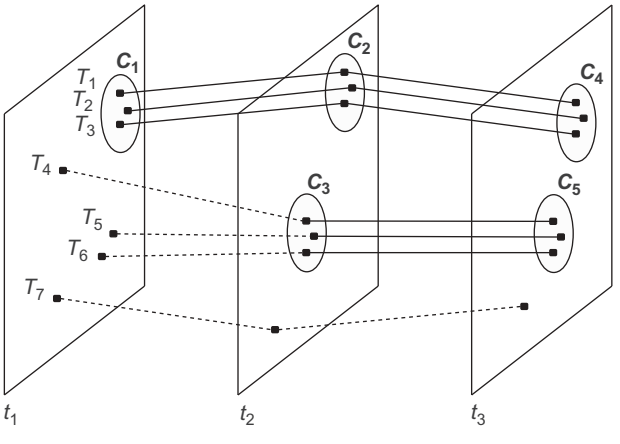

Figure 1. A flock pattern example: T_i illustrates different trajectories, c_i encloses a disk in which trajectories are considered close to one another and t_i represents consecutive time intervals (after Vieira *et al*. 2009).

patterns might also contribute to find moving flock patterns, for example, through the efficient handling of candidates and combinations (Agrawal and Srikant 1994, Han *et al.* 2004).

Another issue of finding flock patterns involves verification and interpretation of those patterns. It is a difficult process and depends on both the characteristics of the object under study and the parameters of the algorithms. The objects might be animals, pedestrians, vehicles or natural phenomena such as hurricanes. The significance of how the objects interact with one another and how they move together will depend on the established knowledge about the objects or phenomena, and accurate interpretation often requires expert knowledge (Gudmundsson *et al.* 2004, Laube *et al.* 2005). But even within the same application domain, flock patterns are context dependent, thus requiring interactive exploration of the input parameters to the algorithm in real time.

In order to be able to work on large data sets and alter the input parameters of the algorithm in real time, the flocking algorithm should be efficient and scalable. So far, results imply that sample size was either too small to test the scalability of the algorithm or relatively high computational complexity and long response times. Therefore, the aim of this article is twofold: (i) to improve the algorithm for mining flock patterns by using a frequent pattern discovery approach and (ii) to provide visual functionality and real-time manipulation of the input parameters of the improved algorithm for verification and interpretation by a domain expert.

2. Related work

Due to the increasing availability of movement data sets, the interest in querying patterns which describe collective behaviour has also increased. Vieira *et al.* (2009) identified three groups of 'collective' patterns in moving object databases: clustering for moving objects, convoy queries and flock patterns. Moving clusters (Jensen *et al.* 2007) and convoy queries (Jeung *et al.* 2008) differ from flock patterns mainly because they do not necessarily contain the same objects during the lifetime of a cluster or convoy. In addition, as these methods are based on spatial clustering, they do not assume any shape restriction.

Previous works in the detection of moving flock patterns were reported by Gudmundsson and van Kreveld (2006) and Benkert *et al.* (2008). They have introduced the use of *disks* with a predefined radius to identify groups of trajectories moving together in the same direction. All trajectories which lie inside of the disk in a particular time instance were considered a candidate pattern. The main limitation of this procedure is that there is an infinite number of possible placements of the disk at any time instance. Indeed, Gudmundsson and van Kreveld (2006) have shown that the discovery of fixed flocks – defined as patterns where the same entities stay together during the entire interval – is an nondeterministic polynomial time-hard problem.

Vieira *et al.* (2009) were the first to present an exact solution for reporting flock patterns in polynomial time and also for those that can work effectively in real time. Their work revealed that a polynomial time solution can be found through identifying a discrete number of locations to place the centre of the flock disk. They proposed the *Basic Flock Evaluation* (BFE) algorithm based on temporal joins and combinations, and four other algorithms based on heuristics, to reduce the total number of candidate disks to be combined and, thus, to reduce the overall computational cost. However, pseudo-code and experimental results still showed relatively high computational complexity, long response time and a large number of discovered flocks, which make interpretation of the results difficult.

Recently, Wachowicz *et al.* (2011) have proposed a new moving flock pattern definition and developed a corresponding algorithm based on the notion of spatio-temporal coherence. The experimental results focused on finding flock patterns in a National Park

in the Netherlands, using high-resolution pedestrian data sets captured using GPS. Although the data capture actual movements, the sample size was too small to test the scalability of this algorithm. Nevertheless, the authors provide an interesting comparison of existing flock detection approaches according to the classification criteria for collective movement recently introduced by Wood and Galton (2009).

Traditional data mining techniques, and particularly the field of FPM, have treated the number of combinations by reducing the number of elements to be combined or compacting the size of the data set. Agrawal and Srikant (1994) have applied pruning techniques based on the downward-closure property, which guarantees that all the subsets of a frequent pattern must also be frequent. Using this property, authors were able to identify invalid candidates and remove them from the analysis. Perhaps the only issue with this technique is that it still scans through the data set repeatedly.

Han and Pei (2000) proposed an intermediate layer which organises the records in a compact data structure called frequent-pattern tree (FP-tree). Main advantages of this methodology include compression of data sets, minimisation of scans and detection of patterns *without* candidate generation (Han and Pei 2000, Borgelt 2005). Recently, Han *et al.* (2004) have proposed a novel and improved FP-tree structure applied in different contexts, for instance, market basket, association rules and sequential patterns. Shan and Wei (2010) have applied this methodology successfully to find co-orientation patterns from satellite imagery. The empirical results show an improvement around one order of magnitude with respect to the traditional approach.

Similarly, an alternative algorithm developed by Uno *et al.* (2005), the Linear time Closed itemset Miner (LCM), has demonstrated a remarkable performance in dense databases using binary decision diagrams, a compact graph-based data structure. Using this approach, frequent patterns can be efficiently processed using algebraic operations. LCM requires linear time to mine frequent patterns when the data compression works well. A comparison performance of LCM and other state-of-the-art techniques is given in Bayardo *et al.* (2004) and Goethals and Zaki (2003).

However, Han and Kamber (2006) show how FPM may generate a huge number of frequent patterns. Specifically, if long records and patterns exist in the data, if a pattern is frequent, each of its subpatterns are frequent as well. This tendency clearly increases the complexity of analysis and understanding. To overcome this problem, closed and maximal pattern mining was proposed (Bayardo 1998, Pasquier *et al.* 1999). The general idea is to report only the longest patterns, avoiding its subpatterns.

The aforementioned techniques have been applied successfully to diverse scenarios such as bioinformatics (Chen *et al.* 2001, Creighton and Hanash 2003), GIS (Han *et al.* 1997, Miller and Han 2001) and marketing (Zhang and Zhang 2002, Giudici and Corporation 2003). Interested readers should refer to Han *et al.* (2007) for a comprehensive survey of the FPM approach. Additionally, the frequent itemset mining implementation (FIMI) repository (Goethals 2004) presents a collection of open source implementations for the most efficient and scalable frequent/closed/maximal pattern mining algorithms.

Overall, the FPM approach has made a tremendous progress in the past decade, and it is thought that this can contribute adequately to solve the drawbacks of finding moving flock patterns in large trajectory data sets. Therefore the main aim of the remainder of this article is to (i) explore a methodology which allows the identification of moving flock patterns using FPM approach, (ii) compare the performance of the original BFE and the one with the FPM implementation and (iii) embed the improved algorithm into a visual environment and provide an interactive interface for changing the algorithm's parameters.

Each of these steps is explained in the remainder of this article.

3. Methodology to include FPM approach into BFE

To build the proposed framework, it was decided to keep the first part of the BFE algorithm, but to address the combinatorial problem using a FPM approach. The first stage in both the approaches involved generation of a final set of disks of trajectory clusters in each timestamp. Furthermore, in order to use FPM to address the combinatorial problem of the BFE, the following three steps are needed:

(1) Construct a transactional version of the trajectory data set based on the disks visited by each trajectory.
(2) Apply an FPM algorithm in the generated database.
(3) Perform postprocessing procedures to check consecutiveness, prune duplicates and report patterns.

Below we describe in details each step of the framework.

3.1. Timestamp-based clustering

The first step of the framework, common to both the approaches, is to identify a final set of clusters in each timestamp. At this stage, both the BFE algorithm and the proposed framework use a fixed disk shape, a circumference with a predefined diameter and an Euclidean distance metric. The main objective is the generation of a final set of disks which cluster the number of trajectories in groups according to proximity criteria. This step still uses the parameter ε, to define the diameter of the disks, and μ, for pruning procedures to reduce the number of valid disks, as proposed in Vieira *et al.* (2009).

3.2. From trajectories to transactions

In a general sense, spatio-temporal data sets comprise information for the location of an entity at a specific time. Each entry in the data set reflects an observation of a point, which in turn describes a specific trajectory. To be able to analyse trends in the data, we assume that spatio-temporal data sets contain at least four fields: a trajectory ID, a timestamp and the X and Y coordinates of the location.

In order to use FPM algorithms, the input database should follow the {*TID:itemset*} schema (Han and Kamber 2006). The ID of the trajectory can be used to identify its corresponding transaction, but it is necessary to define an Item ID which collects information of the time and location of each point. An unique ID is tagged to each disk generated in the first step of the framework. A specific disk will represent a particular region in space and time, and each trajectory can be depicted by the disks that it visits. This concept is illustrated in the following example:

In Figure 1 we can see an example data set consisting of seven trajectories (T_i) that form five disks (c_i). Table 1 is created from the disks which are visited by each trajectory at a specific timestamp (t_i). If Table 1 is treated as a transactional database, it is possible to apply any FPM algorithm to find the frequent patterns. For instance, by setting the minimum support count (min_sup) to be the same value as the minimum number of trajectories μ. If we use $\mu = 3$ the patterns $\{C_1, C_2, C_4 : 3\}$ and $\{C_3, C_5 : 3\}$ should be found. These patterns contain the information about the trajectory members and duration of the possible moving flock patterns.

It is not necessary to have a complete set of all frequent patterns. Only the set of maximal frequent patterns will suffice, since only the longest flock patterns are reported.

Table 1. Transactional version of the data set from Figure 1.

TID	Disk IDs
T_1	$\langle C_1, C_2, C_4 \rangle$
T_2	$\langle C_1, C_2, C_4 \rangle$
T_3	$\langle C_1, C_2, C_4 \rangle$
T_4	$\langle C_3, C_5 \rangle$
T_5	$\langle C_3, C_5 \rangle$
T_6	$\langle C_3, C_5 \rangle$
T_7	$\langle \phi \rangle$

The maximal (or 'closed') sets of frequent patterns avoids the need to set a parameter δ to limit the duration of the flock patterns. In the proposed framework, the parameter δ is used only to set the minimum duration allowed, but flocks with the longest duration will be reported. In contrast, BFE used δ to report flocks with that specific time duration in order to minimise the number of intermediate flocks to be combined. As a result, the final number of flocks reported by the proposed framework is significantly smaller than the number of flocks reported by BFE.

Although the patterns are considered as valid output from FPM algorithms, they will require additional checking before they can be reported as valid flocks (see Section 3.4).

3.3. *FPM algorithms*

Since Agrawal and Srikant (1994) first published their work, improvements and other methods of finding frequent patterns in an efficient and robust way have been proposed. The most popular solution involves the use of compact data structures which compresses the original database, such as FP-trees (Han and Pei 2000, Han *et al.* 2004) and binary decision diagrams (Uno *et al.* 2004, Uno *et al.* 2005, Minato *et al.* 2008). Depending on the application context many implementations and additional variations can be introduced in order to find different types of frequent patterns, such as maximal and closed frequent itemsets.

The FIMI (Goethals 2004) is an initiative that collects the most representative state-of-the-art algorithms, their implementations, their open source code and the sample data sets. It also analyses the performance of different techniques with varying parameters and in relation to different data sets (Goethals 2003, Goethals and Zaki 2004).

Based on the above, LCM (Uno *et al.* 2005) was chosen since it demonstrated a remarkable efficiency using extremely low values of support in sparse data sets. LCM is a backtracking (or depth-first) algorithm based on recursive calls. Here, we omit the detailed description of the algorithm which is described further in Uno *et al.* (2005). LCM has two variants of the program available; LCM_max and LCM_closed will retrieve the maximal or closed set of frequent patterns, respectively. In our case, the output M is a plain text file, where each line is a maximal pattern that contains a set of disk IDs separated by spaces.

3.4. *Postprocessing stage*

Information about time and location for each point of the trajectories was encoded into unique IDs for the disks. Once the LCM algorithm retrieves the set of frequent patterns, it is necessary to decode this information and check the quality and validity of the flocks. It is possible that the members of a valid frequent pattern belong to disks in non-consecutive

times; hence it is necessary to check this requirement, in addition to the minimum duration (δ), before reporting it as a valid flock.

As in the BFE algorithm, the overlapping issue requires the pruning of duplicates and checking for redundant patterns. Using a tree structure, flock patterns with the same members (and the same start and end timestamps) will be easily detected and excluded. Redundant patterns occur when two patterns share exactly the same members but the time duration of one of them is contained by the longest pattern. Using the same data structure, this kind of pattern can also be detected, keeping just the longest one. Once the post-processing stage finishes, the final flock patterns are reported.

4. Comparing performance

An implementation of the BFE algorithm was based on the pseudo-code published in Vieira *et al.* (2009) using Java 1.6. Similarly, a functional prototype of the proposed framework was implemented in Java. The pseudo-code of the proposed framework is presented in Algorithm 1.

Algorithm 1: Computing flocks using a frequent pattern mining approach.
Input: parameters μ, ε and δ, set of points T
Output: flock patterns F

```
1: for each new time instance t_i ∈ T do
2:         C ← call Index.Disks(T[t_i], ε)          // call Algorithm 1 in Vieira et al. (2009)
3:         for each c_i ∈ C do
4:                 P ← c_i.points
5:                 for each p_i ∈ P do
6:                         c_i.time ← t_i
7:                         D[p_i] ← add c_i.id
8:                 end for
9:         end for
10: end for
11: min_sup ← μ
12: M ← call LCM_max(D, min_sup)                    // call LCM Algorithm (Uno et al. 2005)
13: for each max_pattern ∈ M do
14:         id_0 ← max_pattern[0]
15:         c_0 ← C[id_0]
16:         u ← c_0.points
17:         u.t_start ← c_0.time
18:         n ← max_pattern.size
19:         for i = 1 n do
20:                 id_i ← max_pattern[i]
21:                 c_i ← C[id_i]
22:                 if c_i.time = c_{i−1}.time + 1 then
23:                         u ← u∩c_i.points
24:                         u.t_end ← c_i.time
25:                 else
26:                         if u.t_end − u.t_start > δ u ∉ F then
27:                                 F ← add u
28:                                 u.t_start ← c_i.time
29:                         end if
```

```
            end if
        end for
        if u.t_end − u.t_start > δ and u ∉ F then
            F ← add u
        end if
    end for
    return F
```

The results were produced using synthetic and real data sets on an AMD Athlon 64 × 2 dual processor with 3 GB RAM and a 120 GB 7200 RPM hard disk, running Ubuntu Linux 2.6.32. In all cases, experiments ran Java configured with 2048 MB memory.

4.1. *Synthetic data sets*

A group of synthetic data sets were used to test the performance of both the algorithms. First, a synthetic data set was created using the network from San Joaquin County, USA, provided at the Network-based Generator website (Brinkhoff 2011) as is described in Brinkhoff (2002). This data set (SJ50KT55) consists of 50,000 trajectories from 2,014,346 points during 55 timestamps. A second synthetic data set was prepared using the TAPAS Cologne scenario (Varschen and Wagner 2006) in SUMO (Krajzewicz *et al.* 2002), a recognised traffic simulator for urban mobility. The TAPAS Cologne simulation scenario describes the traffic within the city of Cologne (Germany) for a whole day. The main advantage of this data set is that the trajectories are not generated randomly. The original demand data stem from TAPAS, a system which computes mobility wishes for an area population generated based on the information about travelling habits of Germans and the infrastructure of the area they live in (MiD2002 Project 2011). The original data set is huge and only a 2-hour version is publicly available (SUMO Project 2011). Due to memory constrains trajectories shorter than 20 minutes were pruned. The final data set collects 49,225 trajectories and more than 1.8 million points.

Table 2 summarises the main information about both data sets. It is important to clarify that *Trajectory size* refers to the average number of time intervals rather than to average segment length.

4.1.1. *Results*

The BFE algorithm and the proposed framework were evaluated by running them with the SJ50KT55 and TAPAS data sets using different parameters. In the SJ50KT55 data set the diameter of the disk was varied in intervals ranging from 50 to 300 metres. Figure 2a shows the final results. In a similar fashion, the TAPAS Cologne scenario was tested using different ε values (from 20 to 80 metres). The other parameters were set as $\mu = 10$

Table 2. San Joaquin network data sets.

Data set	Network	Number of trajectories	Number of points	Trajectory size (avg)
SJ50KT55	San Joaquin County, USA	50,000	2,014,346	37
TAPAS Cologne	Cologne, Germany	49,225	1,813,454	37

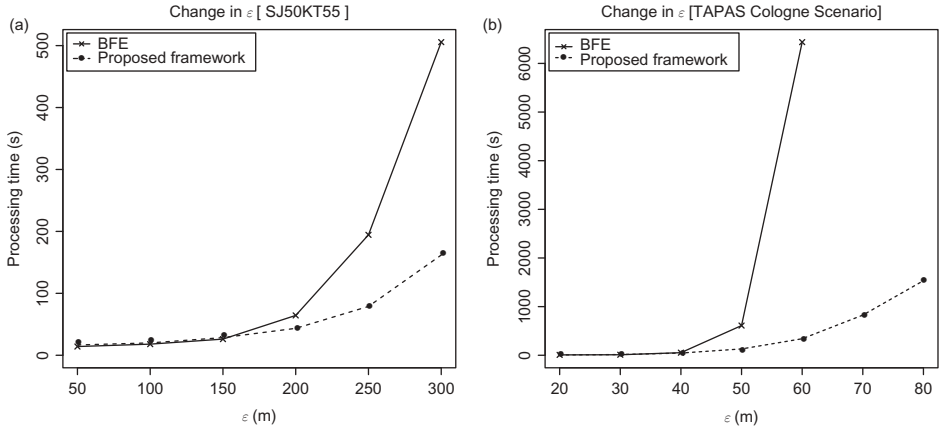

Figure 2. Performance of the BFE algorithm and the proposed framework with different values for ε in the SJ50KT55 and TAPAS Cologne data sets. The additional parameters for SJ50KT55 were set as $\mu = 9$ members, $\delta = 3$ time steps and for TAPAS Cologne were set as $\mu = 10$ members, $\delta = 5$ time steps, respectively.

members and $\delta = 5$ time steps. The results were very similar to that of the San Joaquin network. However in this case, the BFE implementation could not handle ε values greater than 60 metres (it returns an *Out of Memory* exception as can be seen in Figure 2b).

4.2. Real data sets

Besides the synthetic data sets tested above, the proposed framework was also evaluated with trajectories collected from the real world. After some preliminary evaluation, two real data sets from different application domains were selected to test the proposed framework. The first data set tracks iceberg movement in Antarctica using various satellite sensors during 1978 and from 1992 to 2009. The National Ice Centre (NIC) and Brigham Young University (BYU) Microwave Earth Remote Sensing Laboratory have used various satellite sensors to manually track large Antarctic icebergs and collect their positions. Ballantyne and Long (2002) presented a long-term analysis of the Antarctic iceberg activity based on scatterometer and radiometer data. It was decided to apply a linear interpolation on a daily basis and focus on iceberg trajectories during 2006 since they presented the largest amount of records.

The second real data set collects movement information from a group of people around the metropolitan area of Beijing, China. The data set was collected during the Geolife project (Microsoft Research Asia 2011) by 165 anonymous users who used GPS devices over a period of 2 years from April 2007 to August 2009. Locations were recorded by different GPS loggers or smartphones, and most of them have a high sampling rate.

The full data set collects more than 1.2 million of point locations and 23,800 trajectories. However, this data set tracks relatively few moving entities (165 users) in a long time window (more than 2 years) with little temporal overlap. Therefore it was decided to create a modified version by having all of them start at the same time. Again, due to the memory constrains, the trajectories shorter than 10 minutes and longer than 3 hours were pruned. The alternative data set stores 760,814 point locations and 18,216 real trajectories happening together. Table 3 summarises the details for the Iceberg data set and for both the original and alternative Beijing data sets.

Table 3. Iceberg trajectories during 2006 in Antarctica.

Data set	Study area	Number of trajectories	Number of points	Trajectory size (avg)
Icebergs06	Antarctica	49	16,131	329
Original_Beijing	Beijing, China	23,800	1,207,110	50
Modified_Beijing	Beijing, China	18,216	760,814	42

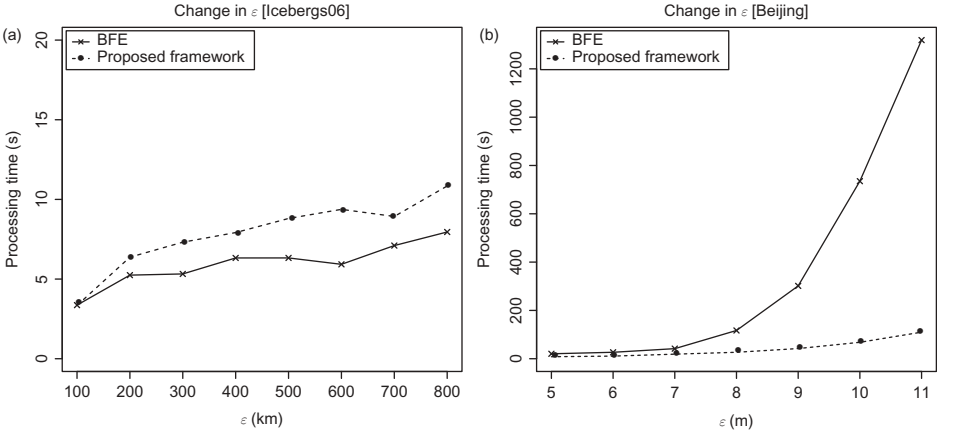

Figure 3. Comparison of both the methods with different values for ε in the Iceberg data sets (a). Modified version for the Beijing data sets with the additional parameters set as $\mu = 5$ members, $\delta = 5$ time steps, respectively (b).

4.2.1. *Results*

With the icebergs data set, a set of tests were performed using both the BFE algorithm and the proposed framework. At this time, the ε parameter was changed to the order of kilometres. The other parameters remain constant at $\mu = 3$ members and $\delta = 3$ time steps. The results of the experiments are shown in Figure 3a. In this case a slightly better performance can be observed by the BFE algorithm. It seems that the larger average trajectory size (329) for this test affects the performance of the FPM algorithm. This issue is addressed further in Section 6. A plot with the comparison between methods for the Beijing data set is shown in Figure 3b. Noticeably, a better performance of the proposed framework is evident, reporting similar behaviour as seen in the results from the synthetic data sets.

5. Embedding the algorithm into a visual environment

The proposed framework is now able to handle large data sets and efficiently find flock patterns. In order to verify those flocks, the visual interface should also be able to deal with the visualisation of these large data sets.

Andrienko *et al.* (2011) offer comprehensive review of the most common ways to visualise the trajectory data and spatio-temporal events. The established visualisation techniques are animated map and Space-Time Cube (STC) (Andrienko *et al.* 2011).

At the faculty of Geo-Information Science and Earth Observation (ITC), University of Twente, the Netherlands, a software implementation of Torsten Hägerstrand's STC was developed, in order to support research in visualisation and analysis of spatio-temporal and multi-attribute data sets (Kraak and Koussoulakou 2004). The objective was to take

full advantage of the capabilities of 3D graphics hardware, that is available on all modern computers, in order to resemble an interactive STC.

Over the past years, the STC's capabilities were extended with a large quantity of options for visualisation of and interaction with the data, in order to support several use-cases. Among others, the current version of the cube can visualise spatio-temporal data as space–time paths, stations, taxels, with attribute-dependent size and colour, inline graphs, text and multimedia annotations. The STC has proven to be useful for the visualisation of movement data (Kraak and Huisman 2009) and could be extended to show spatio-temporal distribution of the resulting flocks, to verify the flock patterns and to represent the actual trajectories that formed those flocks.

To illustrate the functionality of the visual environment we have chosen the real-world data set of tropical cyclones. The data set contains 30 years of tracks from 1980 to 2009, containing 2759 storm tracks with six hourly observations (National Climatic Data Center (NCDC), NOAA 2013). A typical scenario when analysing this data set would be to find the similarity between the tropical cyclones paths. Since tropical cyclones form over the warmer oceans, these flocks would be useful for answering the question *if similarly moving cyclones also share similarity in the environmental factors that influence them?* In order to answer this question we have modified the original data sets so that all cyclones would have a same starting year (1980), but the original month, day and hour were kept intact (Figure 4).

Users have options of manipulating the STC, sliding the temporal plane, switching between 2D (map) and 3D (STC) representations and changing the input parameters for finding flocks. It is implemented in such a way that altering the μ, δ values almost instantly changes the number of resulting flocks on the display (Figure 5). This is because the most computationally intensive part of the algorithm does not have to be repeated when changing those values. Delays in visualisation of the resulting flocks can be observed when increasing the ε value (e.g., changing ε from 300 to 600 km in these data sets introduces delay of about 1–5 seconds depending on the system configuration since the algorithm has to start from the beginning).

The high performance allows for interactive recomputation of the flocks, until desired/optimal values are found. The search for optimal flock patterns is a subjective process that depends on the application domain and the definition of flock in given data sets; there is

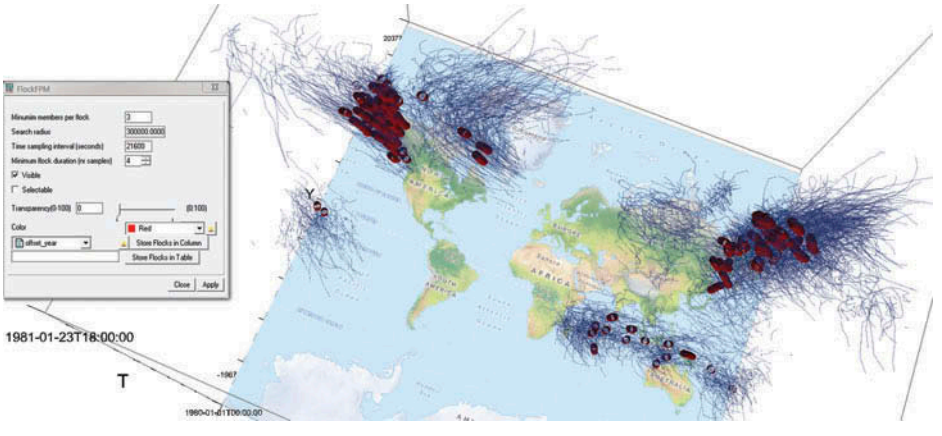

Figure 4. Tropical cyclones and resulting flock patterns with $\varepsilon = 300,000$ map units (300 km), $\mu = 3$ members and $\delta = 4$ time steps, respectively, where each time step being 6 hours.

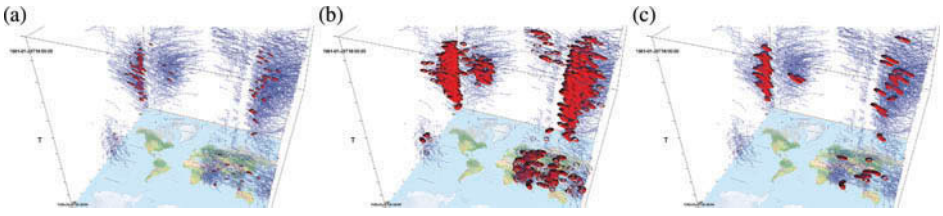

Figure 5. Tropical cyclone flocks with different values of ε = 300 km, δ = 4 (a) and ε = 600 km, δ = 4 (b) and ε = 600 km, δ = 14 (c) time steps. The parameter μ was fixed to three members.

no known computational way for directly finding the input values of the algorithm. For instance, the flocking patterns that are visualised in Figure 5 were determined interactively, and they were found to be representative for the data set and consistent with the findings that indicate that the average size of tropical cyclones varies between 330 and 660 km (Chavas and Emanuel 2010).

Figures 4 and 5 reveal that in the first half of the year (December to May), flocks occur mostly south of the equator, in the Indian and in the Pacific ocean. Majority of flocks however occur in the second half of the year, north of the equator in Eastern and Western Pacific and in the Western Atlantic. Especially in the Western Pacific tropical cyclones tend to follow the same path over the season and form dense flocks (Figure 5).

The most common way to visualise the clustering results is by colouring the display elements representing the clustered items according to their cluster membership and involving graphical summarisation of clusters, such as generation of convex hulls (Andrienko *et al.* 2011). We have used both the approaches for the task of flock verification; flocks are shown as cylinders where the size is proportional to the value of ε parameter. Selection of flocks on screen changes the colouring of the flocks and trajectories to indicate membership. Thus, users have options of selecting flocks, enquiring flock ID and IDs of trajectories that the flocks consist of (Figure 6) and saving the

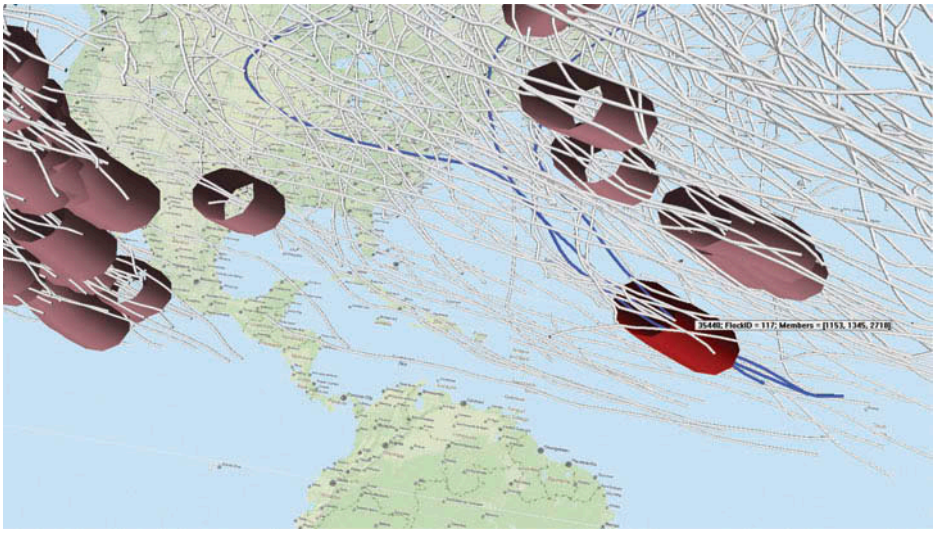

Figure 6. An example of selecting and highlighting flocks with the pop-up window that displays the flock ID and trajectory IDs of its members.

resulting flocks in a column or in a separate table if needed for further analysis and verification (Figure 4). The STC and the newly developed FPM flock detection functionality have been delivered as a plugin for ILWIS, the GIS software that is developed by ITC and 52north. Readers are encouraged to try this functionality by downloading this free and open source software (http://52north.org/communities/ilwis/about).

6. Discussion

6.1. *Impact of trajectory size*

Initial tests with synthetic data sets showed a high performance of the proposed approach with respect to the traditional method. However under tests with the icebergs data set that difference disappeared. It is noted by Bayardo (1998), Grahne and Zhu (2003) and Han *et al.* (2007) that not only the length of the involved transactions but also the length of the resulting patterns have a direct impact in the performance of frequent pattern techniques. The results from the experiments reveal that the shorter the trajectory size, the better the performance of the proposed framework.

Different strategies could be used to limit the size of a trajectory. For example, long periods without change of a position or abrupt jumps in time or location can be used to split long trajectories into shorter segments, without significant loss of spatial information. Similarly, depending on the context, it might be acceptable to interpolate trajectories to longer time intervals. For example, it is possible to obtain a suitable Iceberg data set with samples taken every week instead of daily intervals.

An important feature of the proposed framework is that it is independent of the frequent pattern algorithm. Any technique could use the transactional version of the trajectory data set to retrieve maximal or closed frequent patterns. The issue of mining long transactional data sets has been studied in the bioinformatics field, where *colossal* patterns (from very long transactions) are usually found in huge biological databases such as chromosome sequences, approaching millions of letters. One approach to this problem is mining frequent patterns using a vertical data format, where a relation {*item:TID_set*} is used instead of the traditional {*TID:itemset*} schema Han *et al.* (2007). The CARPENTER (Pan *et al.* 2003) and COBBLER (Pan *et al.* 2004) algorithms are alternatives that follow this format.

6.2. *Overlapping problem*

The use of static values of ε will inevitably generate groups which 'share' trajectories due to the overlapping of disks. This has a negative impact introducing duplicate patterns. Formal definitions of flocks set ε as a fixed parameter (Gudmundsson and van Kreveld 2006, Benkert *et al.* 2008, Vieira *et al.* 2009). However, natural behaviour of moving objects show that groups increase and decrease their members, as well as the extent of the space which they occupy, over time. For example, vehicles move in constrained space with variable speed, which affects the size and shape of the resulting flocks. It seems reasonable to think that flexible shapes and values for ε will model the interaction among moving entities in a more intelligent way.

Spatial clustering algorithms are an option then. These work by discovering set of clusters (instead of disks) of arbitrary shape and size. DBSCAN, Swarm Intelligence techniques (Folino and Spezzano 2002) and grid-based methods (Yang *et al.* 2008) are alternatives to be considered in order to avoid the use of disks with a predefined radius.

6.3. Visualisation

The main purpose of the article was to detect and display flocks instantly by optimising the algorithm and provide users an experience of real-time flock exploration and verification thorough the visual environment. The focus was less on the visualisation functionality that might benefit from such an optimisation. A future direction might include flock visualisation that would involve study of flock composition, recognition of leaders and followers in a particular flock and keeping identity of the same flocks over time.

Although it was easy to convert the improved flocking algorithm the STC plugin of ILWIS, the resulting visualisation might not be the best that is suited for tasks of verifying flocks. More rigorous evaluation is required, preferably in several knowledge domains in order to claim the usefulness. Nevertheless, a combination of the ability to handle large data sets with the possibility to change the input parameters and visually inspect the resulting flocks in real time, at the best of our knowledge, has not been currently implemented or researched.

7. Conclusions

This article used the FPM approach for discovering moving flock patterns in large spatio-temporal data sets. An extended framework which integrates techniques to identify groups of moving entities and the longest duration flocks patterns has been implemented and tested with synthetic and real data sets.

The framework contends that a moving flock pattern can be generalised as a typical frequent pattern. It works by converting a trajectory data set into a transactional database, based on the locations visited by each trajectory. Once a transactional version of the data set is available, FPM algorithms can be applied.

The proposed framework was tested and compared with a widely used existing method (BFE algorithm). Synthetic data sets simulating trajectories generated by large number of moving objects were used to test the scalability of the framework. Real data sets from different application domains and with different characteristics were used to assess the performance and analyse the discovered patterns. Compared with the existing method (BFE), the proposed framework shows better performance in high-dimensional data sets.

The FPM approach showed to be useful to deal with the problems found in the BFE algorithm for data sets with a large numbers of trajectories. The proposed framework is shown to handle the disk combination problem efficiently by using scalable frequent pattern algorithms. Additionally, the flocks which are discovered are ‘cleaner’ since they disregard spurious flocks, while maximal pattern mining techniques are able to detect the longest duration flocks.

The proposed framework is modular, and different techniques can be applied to improve the performance depending on the particular application domain. Although a specific frequent pattern and spatial clustering algorithm was implemented in this framework, alternative methodologies can be used.

The analysis of large spatio-temporal data sets presents significant challenges for data mining in general. FPM techniques have been shown to have important contributions to make in this area. Combining these with the visual interface of the STC within ILWIS allows users to interactively explore large spatio-temporal data sets in search for flocks. This functionality represents a contribution to the field of Geovisual Analytics that is presently unavailable elsewhere.

References

Agrawal, R. and Srikant, R., 1994. Fast algorithms for mining association rules. *In*: J.B. Bocca, M. Jarke, and C. Zaniolo, eds. *VLDB'94*, *Proceedings of the 20th international conference very large data bases*, 12–15 September, Santiago de Chile. Morgan Kaufmann, 487–499. ISBN:1-55860-153-8.

Andrienko, G., *et al*., 2011. A conceptual framework and taxonomy of techniques for analyzing movement. *Journal of Visual Languages & Computing*, 22, 213–232. doi:10.1016/j.jvlc.2011.02.003

Ballantyne, J. and Long, D., 2002. A multidecadal study of the number of Antarctic icebergs using scatterometer data. *In*: *Proceedings of the international geoscience and remote sensing symposium*, Vol. 5, 24–28 June, Toronto, ON. IEEE Publications, 3029–3031.

Bayardo, R., Jr., 1998. Efficiently mining long patterns from databases. *ACM Sigmod Record*, 27, 85–93. doi:10.1145/276305.276313

Bayardo, R., Jr., Goethals, B., and Zaki, M., eds., 2004. FIMI 04, proceedings of the IEEE ICDM workshopon frequent itemset mining implementations. *In: Proceedings of the IEEE ICDM workshop on frequent itemset mining implementations*, 1 November, Brighton. Aachen: CEUR-WS.org, 126.

Benkert, M., *et al.*, 2008. Reporting flock patterns. *Computational Geometry*, 41, 111–125. doi:10.1016/j.comgeo.2007.10.003

Borgelt, C., 2005. An implementation of the FP-growth algorithm. *In*: *Proceedings of the 1st international workshop on open source data mining: frequent pattern mining implementations*, 21–24 August, Chicago, IL. New York: ACM, 5.

Brinkhoff, T., 2002. A framework for generating network-based moving objects. *GeoInformatica*, 6, 153–180. doi:10.1023/A:1015231126594

Brinkhoff, T., 2011. *Network-based generator of moving objects* [online]. Available from: http://iapg.jade-hs.de/personen/brinkhoff/generator/faq.php [Accessed 26 February 2014].

Chavas, D.R. and Emanuel, K.A., 2010. A QuikSCAT climatology of tropical cyclone size. *Geophysical Research Letters*, 37. doi:10.1029/2010GL044558

Chen, R., *et al.*, 2001. Mining association rules in analysis of transcription factors essential to gene expressions. *In*: *Proceedings of the Atlantic symposium on computational biology, and genome information systems & technology*, 15–17 March, Durham, NC.

Creighton, C. and Hanash, S., 2003. Mining gene expression databases for association rules. *Bioinformatics*, 19, 79–86. doi:10.1093/bioinformatics/19.1.79

Dettki, H., Ericsson, G., and Edenius, L., 2004. Real-time moose tracking: an internet based mapping application using GPS/GSM-collars in Sweden. *Alces*, 40, 13–21.

Folino, G. and Spezzano, G., 2002. An adaptive flocking algorithm for spatial clustering. *In*: J.J. Merelo Guervós, *et al*., eds. *Parallel problem solving from nature PPSN VII*, 7–11 September, Granada. Springer, 924–933. ISBN:978-3-540-45712-1 (Online).

Frank, A., Raper, J., and Cheylan, J., 2001. *Life and motion of socio-economic units*. London: Taylor & Francis.

Giudici, P. and Corporation, E., 2003. *Applied data mining: statistical methods for business and industry*. New York: Wiley.

Goethals, B., 2003. *Survey on frequent pattern mining*. Manuscript, 1–43. Available from: http://win.ua.ac.be/~adrem/bibrem/pubs/fpm_survey.pdf [Accessed 21 February 2014].

Goethals, B., 2004. *The Fimi'04 homepage*. Available from: http://fimi.cs.helsinki.fi/ [Accessed 21 February 2014].

Goethals, B. and Zaki, M., eds., 2003. FIMI 03. *Proceedings of the ICDM 2003 workshop on frequent itemset mining implementations*, 19 December, Melbourne, FL, 90. CEUR Workshop Proceedings.

Goethals, B. and Zaki, M., 2004. Advances in frequent itemset mining implementations: report on FIMI'03. *ACM SIGKDD Explorations Newsletter*, 6, 109–117. doi:10.1145/1007730.1007744

Grahne, G. and Zhu, J., 2003. High performance mining of maximal frequent itemsets. *In*: *Proceedings of the 6th international workshop on high performance data mining*, May, San Francisco, CA.

Gudmundsson, J. and van Kreveld, M., 2006. Computing longest duration flocks in trajectory data. *In*: S. Nittel, ed. *Proceedings of the 14th annual ACM international symposium on Advances in geographic information systems*, 10–11 November, Arlington, VA. New York: ACM, 42.

Gudmundsson, J., Van Kreveld, M., and Speckmann, B., 2004. Efficient detection of motion patterns in spatio-temporal data sets. *In*: *Proceedings of the 12th annual ACM international workshop on geographic information systems*, 12–13 November, Arlington, VA. New York: ACM, 250–257.

Han, J., *et al*., 2007. Frequent pattern mining: current status and future directions. *Data Mining and Knowledge Discovery*, 15, 55–86. doi:10.1007/s10618-006-0059-1

Han, J. and Kamber, M., 2006. *Data mining: concepts and techniques*. Waltham, MA: Morgan Kaufmann.

Han, J., Koperski, K., and Stefanovic, N., 1997. GeoMiner: a system prototype for spatial data mining. *In*: J. Peckham, ed. *Proceedings of the 1997 ACM SIGMOD international conference on Management of data*, 13–15 May, Tucson, AZ. ACM Press, 553–556.

Han, J. and Pei, J., 2000. Mining frequent patterns by pattern-growth: methodology and implications. *ACM SIGKDD Explorations Newsletter*, 2, 14–20. doi:10.1145/380995.381002

Han, J., *et al.*, 2004. Mining frequent patterns without candidate generation: a frequent-pattern tree approach. *Data Mining and Knowledge Discovery*, 8, 53–87. doi:10.1023/B:DAMI.0000005258.31418.83

Iwase, S. and Saito, H., 2002. Tracking soccer player using multiple views. *In*: *Proceedings of the IAPR workshop on machine vision applications (MVA02)*, 11–13 December, Nara, 102–105. ISBN:4-901122-02-9.

Jensen, C., Lin, D., and Ooi, B.-C, eds., 2007. Continuous clustering of moving objects. *IEEE Transactions on Knowledge and Data Engineering*, 19 (9), 1161–1174. doi:10.1109/TKDE.2007.1054

Jeung, H., *et al.*, 2008. Discovery of convoys in trajectory databases. *Proceedings of the VLDB Endowment*, 1, 1068–1080.

Kalnis, P., Mamoulis, N., and Bakiras, S., 2005. On discovering moving clusters in spatio-temporal data. *In*: C.B. Medeiros, M.J. Egenhofer, and E. Bertino, eds. *Proceedings of the 9th international symposium on Advances in spatial and temporal databases*, SSTD 2005, 22–24 August, Angra dos Reis. Springer, 364–381. ISBN:978-3-540-28127-6 (Print), 978-3-540-31904-7 (Online).

Kraak, M. and Huisman, O., 2009. Beyond exploratory visualization of space time paths. *In*: H.J. Miller and J. Han, eds. *Geographic data mining and knowledge discovery*. 2nd ed. London: Taylor & Francis, 431–443.

Kraak, M. and Koussoulakou, A., 2004. A visualization environment for the space – time cube. *In*: P. Fisher, ed., *Proceedings of the SDH 2004: proceedings of the 11th international symposium on spatial data handling: advances in spatial data handling II*, 23–25 August, Zurich. Berlin: Springer, 189–200.

Krajzewicz, D., *et al.*, 2002. SUMO (Simulation of Urban MObility). *In*: A. Al-Akaidi, ed. *Proceedings of the 4th Middle East symposium on simulation and modelling*, September, Sharjah. Erlangen: SCS European Publishing House, 183–187.

Laube, P., Kreveld, M., and Imfeld, S., 2005. Finding REMO – detecting relative motion patterns in geospatial lifelines. *In*: P.F. Fisher, ed. *Developments in spatial data handling*. Berlin: Springer-Verlag, 201–215.

Makris, D. and Ellis, T., 2002. Path detection in video surveillance. *Image and Vision Computing*, 20, 895–903. doi:10.1016/S0262-8856(02)00098-7

Microsoft Research Asia, 2011. *GeoLife Gps Trajectories*. Available from: http://research.microsoft.com/en-us/downloads/b16d359d-d164-469e-9fd4-daa38f2b2e13/ [Accessed 21 February 2014].

Mid 2002 Project, 2011. *Mobility in Germany 2002*. Available from: http://daten.clearingstelle-verkehr.de/196/ [Accessed 21 February 2014].

Miller, H. and Han, J., 2001. *Geographic data mining and knowledge discovery*, Vol. 338. Boca Raton, FL: Wiley Online Library.

Minato, S., Uno, T., and Arimura, H., 2008. LCM over ZBDDS: fast generation of very large-scale frequent itemsets using a compact graph-based representation. *In*: T. Washio, *et al*., eds. *Advances in knowledge discovery and data mining*, Berlin: Springer-Verlag, 234–246.

National Climatic Data Center (NCDC), NOAA, 2013. *International Best Track Archive for Climate Stewardship (IBTrACS)*. Available from: http://www.ncdc.noaa.gov/oa/ibtracs/index.php?name=ibtracs-data-access [Accessed 21 February 2014].

Pan, F., *et al.*, 2003. CARPENTER: finding closed patterns in long biological datasets. *In*: L. Getoor, *et al*., eds. *Proceedings of the ninth ACM SIGKDD international conference on knowledge discovery and data mining*, 24–27 August, Washington, DC. New York: ACM, 637–642.

Pan, F., *et al.*, 2004. COBBLER: combining column and row enumeration for closed pattern discovery. *In*: *Proceedings of the scientific and statistical database management, 2004. 16th international conference*, 21–23 June, Santorini Island. Washington, DC: IEEE Computer Society, 21–30. ISBN:0-7695-2146-0.

Pasquier, N., *et al*., 1999. Discovering frequent closed itemsets for association rules. *In*: C. Beeri and P. Buneman, eds. *Database Theory – ICDT'99*, Lecture Notes in Computer Science 1540. Berlin: Springer, 398–416. Available from: http://link.springer.com/chapter/10.1007/3-540-49257-7_25

Piciarelli, C., Foresti, G., and Snidara, L., 2006. Trajectory clustering and its applications for video surveillance. *In*: *Proceedings of the advanced video and signal based surveillance, 2005. AVSS 2005. IEEE conference*, 15–16 September, Como. Washington, DC: IEEE Computer Society, 40–45.

Shan, M. and Wei, L., 2010. Algorithms for discovery of spatial co-orientation patterns from images. *Expert Systems with Applications*, 37, 5795–5802. doi:10.1016/j.eswa.2010.02.028

Sumo Project, 2011. *TAPAS Cologne scenario*. Available from: http://sourceforge.net/apps/mediawiki/sumo/index.php?title=Data/Scenarios/TAPASCologne [Accessed 21 February 2014].

Uno, T., Kiyomi, M., and Arimura, H., 2004. LCM ver. 2: efficient mining algorithms for frequent/closed/maximal itemsets. *In*: *Proceedings of the IEEE ICDM04 workshop FIMI04 (International conference on data mining, frequent itemset mining implementations)*, 1 November, Brighton.

Uno, T., Kiyomi, M., and Arimura, H., 2005. Lcm ver. 3: collaboration of array, bitmap and prefix tree for frequent itemset mining. *In*: *Proceedings of the 1st international workshop on open source data mining: frequent pattern mining implementations*, 21–24 August, Chicago, IL. New York: ACM, 77–86.

Varschen, C. and Wagner, P. 2006, *Mikroskopische Modellierung der Personenverkehrsnachfrage auf Basis von Zeitverwendungstagebüchern*. Vortrag auf dem, 7.

Vieira, M., Bakalov, P., and Tsotras, V. 2009, On-line discovery of flock patterns in spatio-temporal data. *In*: D. Agrawa, *et al*., eds. *Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems*, 4–6 November, Seattle, WA. New York: ACM, 286–295.

Wachowicz, M., *et al.*, 2011. Finding moving flock patterns among pedestrians through collective coherence. *International Journal of Geographical Information Science*, 25, 1849–1864. doi:10.1080/13658816.2011.561209

Wood, Z. and Galton, A., 2009. A taxonomy of collective phenomena. *Applied Ontology*, 4, 267–292.

Yang, Y., Zhang, J., and Yang, J., 2008. Grid-based hierarchical spatial clustering algorithm in presence of obstacle and constraints. *In*: J. Ni, *et al*., eds. *Proceedings of the 2008 international conference on internet computing in science and engineering*, 28–29 January, Harbin. Seattle, WA: IEEE Computer Society, 383–388.

Zhang, C. and Zhang, S., 2002. *Association rule mining: models and algorithms*. Berlin: Springer-Verlag.