2009 IEEE International Conference on Data Mining Workshops

Accurate Discovery of Valid Convoys from Moving Object Trajectories

Hyunjin Yoon
Computer Science Department
University of Southern California
Los Angeles, CA 90089-0781
hjy@usc.edu

Cyrus Shahabi
Computer Science Department
University of Southern California
Los Angeles, CA 90089-0781
shahabi@usc.edu

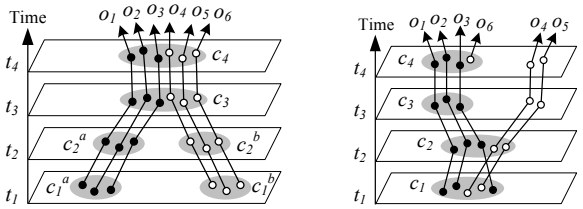
***Abstract*—Given a set of moving object trajectories, it is of interest to find a group of objects, called a *convoy*, that are spatially density-connected for a certain duration of time. However, existing convoy discovery algorithms have a critical problem of accuracy; they tend to both miss *larger* convoys and retrieve *invalid* ones where the density-connectivity among the objects is not completely satisfied. We propose a new valid convoy discovery algorithm, called VCoDA, for the accurate discovery of *valid* convoys from moving object trajectories. Specifically, VCoDA first retrieves all partially connected convoys while guaranteeing no false dismissal of any valid convoys and then validates their density-connectivity to eventually obtain a complete set of valid convoys. Our extensive experiments on three real-world datasets demonstrate the effectiveness of our technique; VCoDA improves the precision by a factor of 3 on average and the recall by up to 2 orders of magnitude as compared to an existing method.**

I. Introduction

A *moving object trajectory* is a series of locations sampled at discrete instances of time. Various types of trajectory data tracking the movement of vehicles, hurricanes, or animals have been acquired using location-aware sensors and exploited to obtain insights from the intrinsic movement traits and behaviors presented in the data [1].

Given a set of moving object trajectories, it is of interest to find a group of objects that moved together for a certain duration of time [2], [3], [4], [5]. Such groups of objects *close* both in time and space dimensions are particularly useful to abstract the accumulated mobilities common in groups. Various group patterns have been defined over moving object trajectories. A (long duration) *flock* [6], [2] is defined by at least m moving objects staying together within a circular region of radius ε during at least k consecutive timestamps. Although the flock pattern has been most popularly exploited in the past, the shape and the size of flock snapshot is limited to a disk of a fixed size bound ε, hence it cannot cover *larger* flocks where objects are distributed over a wider area larger than the given disk size. To avoid this rigid restriction, a variant of flock called a convoy [3] is recently proposed based on the notion of *density-connectivity*. A convoy is defined as a group of at least m moving objects that are *density-connected* w.r.t. the density constraints during at least k consecutive timestamps. Unlike flock, the shape and the size of convoy snapshot can be arbitrary.

Jeung *et al.* first defined the convoy query in [7] and proposed several algorithms to discover all convoys from a given trajectory dataset [7], [3]. Their solution adopts the well-known *moving cluster* algorithm [4] in that a density-based clustering (e.g., DBSCAN [8]) is first performed on the moving objects at each timestamp to find snapshot density-connected clusters of arbitrary shapes and then the intersection of a sequence of at least k such snapshot clusters appearing during consecutive timestamps is detected as a convoy if they share at least m objects in common.



(a) Missed convoy (b) Invalid convoy

Figure 1. Accuracy problems of current convoy discovery algorithms

Our key observation is that current convoy discovery algorithms have a critical problem of *accuracy* in terms of both precision and recall; they tend to miss *larger* convoys and retrieve *invalid* ones where the density-connectivity among the objects is not completely satisfied. First consider the six moving objects o_1,o_2,. . .,o_6 in Figure 1(a). Suppose the minimum number of objects and the *lifetime* of convoys to be mined is set to m=3 and k=2. We can clearly see that the six objects start moving in two groups from t_1 and then travel all together from t_3, which results in three natural group patterns: P_1=$\langle\{o_1,o_2,o_3\},[t_1,t_4]\rangle$, P_2=$\langle\{o_4,o_5,o_6\},[t_1,t_4]\rangle$, and P_3=$\langle\{o_1,o_2,\ldots,o_6\},[t_3,t_4]\rangle$. However, current convoy discovery algorithms are unable to detect the larger convoy P_3 of all six objects. Similarly, the current approach will return only one convoy of three objects P_4=$\langle\{o_1,o_2,o_3\},[t_1,t_4]\rangle$, not the larger group of five P_5=$\langle\{o_1,o_2,\ldots,o_5\},[t_1,t_2]\rangle$, in Figure 1(b). Another problem is that the returned convoy P_4 is just a *partially connected* group since the three (black) objects are not density-connected at t_1, i.e., they are unable to form a cluster by themselves without the other two (white)

978-0-7695-3902-7/09 $26.00 © 2009 IEEE
DOI 10.1109/ICDMW.2009.71

objects. Hence, P_4 should have had a shorter lifetime $[t_2, t_4]$ to be indeed a correct valid convoy.

In this paper, we propose a new algorithm for the accurate discovery of valid convoys from a set of moving object trajectories, named VCoDA (Valid Convoy Discovery Algorithm). Our solution consists of two phases; first a set of all partially connected convoys is discovered from a given set of moving objects and then the density-connectivity of each partially connected convoys is validated to finally obtain a complete set of valid convoys. The first phase of VCoDA extends the current convoy algorithm CMC [7] in that, scanning through the entire span of time, a set of density-connected snapshot clusters are incrementally updated by consecutive ones with sufficient objects in common under four operations (i.e., insert, extend, delete, and return). This approach is further extended in the second phase such that the density-connectivity of each partially connected convoy is incrementally verified at every timestamp either by immediate, single re-clustering, or recursive validation. Our experiments on three real-world datasets demonstrate the effectiveness of our techniques; VCoDA improves the precision by a factor of 3 on average and the recall by up to 2 orders of magnitude as compared to the existing convoy algorithm.

The remainder of this paper is organized as follows. Section II introduces some basic definitions and the problem considered in this paper. Section III proposes our solution for the valid convoy discovery. Section IV presents the results of experimental evaluation. The related work is discussed in Section V, followed by conclusion in Section VI.

II. Preliminaries

The *trajectory* of a moving object is a finite sequence of sampled locations during a closed time interval $[t_1, t_n]$ and defined as a sequence of pairs, $\langle(p_1, t_1),(p_2, t_2),\ldots,(p_n, t_n)\rangle$, where $p_i \in \Re^d$ $(d \in \{2,3\})$ is a two- or three-dimensional vector[1] representing the geo-spatial position sampled at a timestamp $t_i \in \Im^+$ [9]. Figure 2 shows the trajectories of five moving objects as solid *directed* polylines in the *spatio-temporal space* formed by the spatial plane of X and Y axes and the time dimension of Z axis. For simplicity, we use the notation $o(t_i)$ to denote the *snapshot* position p_i of a moving object o sampled at a timestamp t_i.

We now adopt the notions of density-based clustering originally proposed for the algorithm DBSCAN [8] specifically for the snapshot locations of moving objects. Given a set $P=\{o_1(t),\ldots,o_N(t)\}$ of snapshot locations of N moving objects at a timestamp t and a distance threshold ε, the *ε-neighborhood* $N_\varepsilon(o_p(t))$ of a location $o_p(t) \in P$ is defined as $N_\varepsilon(o_p(t))=\{o_q(t) \in P | D(o_p(t), o_q(t)) \leq \varepsilon\}$, where $D(\cdot)$ is the Euclidean distance. A snapshot location $o_p(t)$ is *directly density-reachable* from a location $o_q(t)$ w.r.t. a given distance threshold ε and an integer m if $o_p(t) \in N_\varepsilon(o_q(t))$ and $|N_\varepsilon(o_q(t))| \geq m$. A location $o_p(t)$ is *density-reachable* from a location $o_q(t)$ w.r.t. a given distance threshold ε and an integer m if there is a chain of locations $o_1(t),\ldots,o_n(t) \in P$ s.t. $o_1(t)=o_q(t)$, $o_n(t)=o_p(t)$, and $o_{i+1}(t)$ is directly density-reachable from $o_i(t)$ w.r.t. ε and m for $1 \leq i < n$. A location $o_p(t) \in P$ is *density-connected* to a location $o_q(t) \in P$ w.r.t. a given distance threshold ε and an integer m if there is a location $o_r(t) \in P$ s.t. both $o_p(t)$ and $o_q(t)$ are density-reachable from $o_r(t)$ w.r.t. ε and m. For example, in Figure 2, the snapshot location $o_1(t_1)$ of an object o_1 at a timestamp t_1 is not directly density-reachable from the snapshot location $o_2(t_1)$ of an object o_2, but density-reachable due to a chain of locations $o_2(t_1), o_5(t_1), o_4(t_1), o_1(t_1)$.

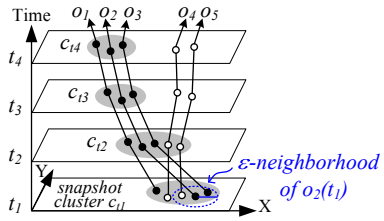


Figure 2. Valid convoys from trajectories

Definition 1. Given a set of moving objects $\mathcal{O}$, a distance threshold ε, and an integer m, a *snapshot cluster* c_t at a timestamp t is a non-empty subset of objects $\mathcal{O}' \subseteq \mathcal{O}$ satisfying the following conditions:
1) Connectivity: $\forall o_p, o_q \in \mathcal{O}'$, a location $o_p(t)$ is density-connected to a location $o_q(t)$ w.r.t. ε and m.
2) (Spatial) maximality: $\forall o_p, o_q \in \mathcal{O}'$, if $o_q \in \mathcal{O}'$ and a location $o_p(t)$ is density-reachable from a location $o_q(t)$ w.r.t. ε and m, then also $o_p \in \mathcal{O}'$.
3) Sufficient objects: $|\mathcal{O}'| \geq m$.

A snapshot cluster is a group of density-connected objects with arbitrary shape and size yet constrained to a single timestamp. Figure 2 shows four snapshot clusters discovered at each timestamp with the parameter m=3. Such snapshot clusters are *spatially maximal* such that no two snapshot clusters at a timestamp can overlap in their objects. We extend this notion of density-based *spatial* clusters to that of density-based *spatio-temporal* clusters for moving objects.

Definition 2. Given a set of moving objects $\mathcal{O}$, a distance threshold ε, an integer m, and a lifetime integer k, a *valid convoy* is a non-empty subset of objects $\mathcal{O}' \subseteq \mathcal{O}$ during a time interval $[t_a, t_b]$, satisfying the following conditions:
1) Connectivity: $\forall o_p, o_q \in \mathcal{O}'$, a snapshot location $o_p(t)$ is density-connected to a snapshot location $o_q(t)$ w.r.t. ε and m, which holds for every timestamps t, $t_a \leq t \leq t_b$.
2) Spatial maximality: $\forall o_p, o_q \in \mathcal{O}'$, if $o_q \in \mathcal{O}'$ and $o_p(t)$ is density-reachable from $o_q(t)$ for all $t_a \leq t \leq t_b$, then $o_p \in \mathcal{O}'$

[1]For the purpose of proper visualization, we assume two-dimensional location vectors throughout this paper.

3) Temporal maximality: $\neg\forall o_p, o_q \in \mathcal{O}'$, $o_p(t_a-1)$ is density connected to $o_q(t_a-1)$ at t_a-1 ($t_a>1$) and $\neg\forall o_r, o_s \in \mathcal{O}'$, $o_r(t_b+1)$ is density-connected to $o_s(t_b+1)$ at t_b+1 ($t_b<n$).
4) Sufficient objects: $|\mathcal{O}'| \geq m$.
5) Sufficient lifetime: $(t_b - t_a + 1) \geq k$.

A valid convoy is therefore a group of density-connected objects with arbitrary shape and extent during a sufficient consecutive time interval. Such valid convoys are *spatially* and *temporally maximal* such that no two valid convoys with the same time interval can overlap in their objects and no two valid convoys with the same set of moving objects can overlap in time. We use the notation v=$\langle\{o_1,\ldots,o_n\},[t_a,t_b]\rangle$ to denote a valid convoy v consisting of moving objects $o_1,\ldots,o_n$ that are thoroughly density-connected during a consecutive time interval $[t_a, t_b]$. A valid convoy $\langle\{o_1,o_2, o_3\},[t_2, t_4]\rangle$ can be found over the five trajectories in Figure 2, with the parameters m=3 and k=3.

Definition 3. Given a set of moving objects $\mathcal{O}$, a distance threshold ε, an integer m, a lifetime integer k, and a sequence of snapshot clusters $c_{t_a},c_{t_a+1},\ldots,c_{t_b}$ during a consecutive time interval $[t_a, t_b]$, a *partially density-connected convoy* (or simply partially connected convoy) is defined as a non-empty subset of objects $\mathcal{O}' \subseteq \mathcal{O}$ during a time interval $[t_a, t_b]$, satisfying following conditions:
1) Spatial maximality: $\mathcal{O}' = c_{t_a} \cap c_{t_a+1} \cap \ldots \cap c_{t_b}$.
2) Temporal maximality: $\nexists c_{t_a-1}$, c_{t_b+1}, $\mathcal{O}' \subseteq c_{t_a-1}$ and $\mathcal{O}' \subseteq c_{t_b+1}$
3) Sufficient objects: $|\mathcal{O}'| \geq m$.
4) Sufficient lifetime: $(t_b - t_a + 1) \geq k$.

In other words, a partially connected convoy is a group of objects that traverse in a sequence of dense regions during a consecutive time interval of sufficient lifetime. Although the objects are within a dense area at each timestamp, they are not necessarily density-connected by themselves unlike valid convoys. For example, a partially connected convoy $\langle\{o_1,o_2, o_3\},[t_1, t_4]\rangle$ is found over the five moving object trajectories in Figure 2, with the parameters m=3 and k=3. As can be seen, the three objects o_1,o_2,o_3 are not density-connected by themselves at the timestamp t_1 without the objects o_4 and o_5 but are lying in a dense region corresponding to the snapshot cluster c_{t_1}, which is therefore not a valid convoy due to the unsatisfied connectivity constraint.

Definition 4. Given two valid (or partially connected) convoys v=$\langle\mathcal{O}, [t_a, t_b]\rangle$ and v'=$\langle\mathcal{O}', [t_{a'}, t_{b'}]\rangle$, v is a *sub-convoy* of v', denoted as v=*sub-convoy*(v'), if either one of the following conditions is satisfied exclusively:
1) $\mathcal{O} = \mathcal{O}'$ and $t_{a'} \leq t_a \leq t_b \leq t_{b'}$, or
2) $\mathcal{O} \subseteq \mathcal{O}'$ and $t_{a'} = t_a \leq t_b = t_{b'}$.

Given a set of moving objects $\mathcal{O}$, a distance threshold ε, an integer m, and an integer k, the convoy discovery problem is to mine all valid convoys from $\mathcal{O}$, each consisting of at least m moving objects that are density-connected w.r.t. the density constraints m and ε during at least k consecutive timestamps.

III. Discovering Valid Convoys

In this section, we present our solution VCoDA for the valid convoy discovery problem. First, a set of all possible partially connected convoys is discovered from a given set of moving object trajectories in Section III-A. Then, the density-connectivity of each partially connected convoy is validated to obtain a set of truly valid convoys satisfying all five constraints in Section III-B.

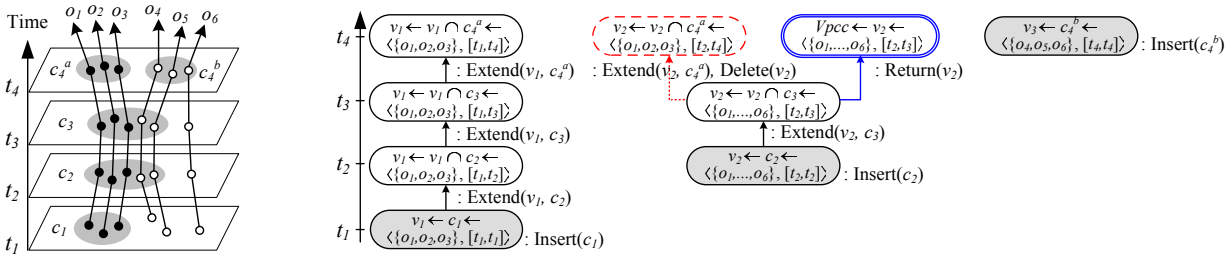
A. Partially Connected Convoy Discovery

Our approach of finding a partially connected convoy extends the well-known *moving cluster* algorithm [4] as in other convoy discovery algorithms [3]. First, we perform a density-based clustering DBSCAN [8] on the *snapshot* locations of moving objects at each timestamp to identify the snapshot clusters defined in Definition 1. Figure 3(a) shows such snapshot clusters discovered at each timestamp with the parameter m=3. Starting with the set $\mathcal{C}$ of snapshot clusters at the first timestamp, we incrementally update a set $\mathcal{V}$ of *current* partially connected convoy candidates scanning throughout the timestamps. The set $\mathcal{V}$ is maintained at each timestamp under four operations defined in Table I: 1) insertion of a snapshot cluster $c \in \mathcal{C}$ as a new partially connected convoy candidate, 2) extension of an existing partially connected convoy candidate $v \in \mathcal{V}$ by a *matching* snapshot cluster $c \in \mathcal{C}$, 3) deletion of a *sub-convoy* $v \in \mathcal{V}$, and 4) return of a *maximally* extended partially connected convoy candidate $v \in \mathcal{V}$ to an actual partially connected convoy.

Figure 3(b) shows all possible updates of $\mathcal{V}$ during a time interval $[t_1, t_4]$, with the parameters m=3 and k=2. At t_1, the cluster c_1 is inserted to $\mathcal{V}$ as a new partially connected convoy candidate v_1 by the first condition of Insert() in Table I since $\mathcal{V}$ is initialized as an empty set. At t_2, the current partially connected convoy candidate v_1 is compared with the snapshot cluster c_2, resulting in an extended convoy candidate consisting of their intersection o_1, o_2, and o_3. Although matched to the convoy candidate v_1, the snapshot cluster c_2 should be also inserted to $\mathcal{V}$ as a new partially connected convoy candidate since it is not fully absorbed by any of the candidates in $\mathcal{V}$ (see the second condition of Insert() in Table I). At t_3, both v_1 and v_2 are extended by the cluster c_3. At t_4, the convoy candidate v_1 is extended by the matching cluster c_4^a. The convoy candidate v_2 is also extended by c_4^a, which however leads to a sub-convoy of the extended convoy candidate v_1. Therefore, this extended partially connected convoy candidate is deleted from the set $\mathcal{V}$. In addition, v_2 must be returned as a partially connected convoy satisfying the constrains m=3 and k=2 at the current timestamp t_4 since is not absorbed by any of the snapshot clusters in $\mathcal{C}$. At last, the snapshot cluster c_4^b is inserted to

Table I
FOUR OPERATIONS TO UPDATE $\mathcal{V}$ IN THE PARTIALLY CONNECTED CONVOY DISCOVERY

Operation	Conditions
Insert(c)	(1) c is *not matched*: $\nexists v \in \mathcal{V}$, $\lvert v \cap c \rvert \geq m$, or (2) c is matched but *not absorbed*: $\exists v \in \mathcal{V}$, $\lvert v \cap c \rvert \geq m \wedge c \backslash (v \cap c) \neq \emptyset \wedge \nexists v' \in \mathcal{V} \backslash \{v\}$, v'=c
Extend(v,c)	v and c share enough objects: $\lvert v \cap c \rvert \geq m$
Delete(v)	v is a sub-convoy: $\exists v' \in \mathcal{V}$, $v = \textit{sub-convoy}(v')$ (by Def. 4)
Return(v)	(1) v is *not extended*: $\nexists c \in \mathcal{C}$, $\lvert v \cap c \rvert \geq m \wedge v$.lifetime $\geq k$, or (2) v is extended but *not absorbed*: $\exists c \in \mathcal{C}$, $\lvert v \cap c \rvert \geq m \wedge v$.lifetime$\geq k \wedge v \backslash (v \cap c) \neq \emptyset \wedge \nexists c' \in \mathcal{C} \backslash \{c\}$, v=c'



(a) Snapshot clusters

(b) Updates of the current partially connected convoy candidates $\mathcal{V}$

Figure 3. Discovery process of partially connected convoys

$\mathcal{V}$ as a new partially connected convoy candidate since it is not matched to any of the convoy candidates.

Algorithm 1 PCCD(moving objects $\mathcal{O}$, ε, m, k)

```
1: V ← ∅; // set of current partially connected convoys
2: for each timestamp t do
3:    V_next ← ∅; // next set of partially connected convoys
4:    C ← DBSCAN(P_t(O), m, ε);
5:    for each cluster c ∈ C do // initialize snapshot clusters
6:       c.matched ← false; c.absorbed ← false; c.lifetime ← [t,t];
7:    for each current convoy candidate v ∈ V do
8:       v.extended ← false;   v.absorbed ← false;
9:       for each cluster c ∈ C do
10:         if |v ∩ c| ≥ m then // Extend(v, c)
11:            v.extended ← true;   c.matched ← true;
12:            v_ext ← ⟨v ∩ c, [v.lifetime_start, t]⟩;
13:            V_next ← updateVnext(V_next, v_ext);
14:            if v ⊂ c then v.absorbed ← true;
15:            if c ⊂ v then c.absorbed ← true;
16:      if not v.extended or not v.absorbed then
17:         if v.lifetime ≥ k then  V_pcc ← V_pcc ∪ {v} // Return(v) ;
18:   for each cluster c ∈ C do
19:      if not c.matched or not c.absorbed then
20:         V_next ← updateVnext(V_next, c); // Insert(c)
21:   V ← V_next;
22: return  V_pcc;
```

Algorithm 1 shows the pseudocode of our partially connected convoy discovery algorithm (PCCD). As an input, it takes trajectories of moving objects $\mathcal{O}$, a distance threshold ε, the minimum number of objects m, and the minimum lifetime k. At each timestamp, the snapshot clusters $\mathcal{C}$ satisfying the density constraints (ε and m) are first obtained in line 4 and compared with the current partially connected convoy candidates in $\mathcal{V}$ (lines 7 and 9). If a current convoy candidate $v \in \mathcal{V}$ shares at least m objects with a cluster $c \in \mathcal{C}$ (line 10), then v is extended with c into a v_{ext} as in line 12. Now, we call the function updateVnext() to update the set of *next* partially connected convoy candidates $\mathcal{V}_{next}$ with this v_{ext} so as to be used in the next iteration (line 13). Then, v and c should be compared again if one is absorbed by the other in lines 14-15. The current convoy candidates in $\mathcal{V}$ that are neither extended nor absorbed by any of clusters in C and have the lifetime of at least k are returned to the set $\mathcal{V}_{pcc}$ of actual partially connected convoys in lines 16-17, which is the output of our PCCD algorithm. Similarly, the current snapshot clusters in $\mathcal{C}$ that are neither matched nor absorbed by any of the current partially connected convoy candidates in $\mathcal{V}$ are inserted to the next partially connected convoy candidates $\mathcal{V}_{next}$ in lines 18-20, to be used in the next iteration.

Algorithm 2 updateVnext($\mathcal{V}_{next}$, v_{new})

```
1: if ∃v ∈ V_next s.t. v = v_new then
2:    if v.lifetime_start > v_new.lifetime_start then // v is a sub-convoy
      of v_new
3:       V_next ← V_next \ {v}; // Delete(v)
4:       V_next ← V_next ∪ {v_new};
5: else
6:    V_next ← V_next ∪ {v_new};
7: return  V_next;
```

Algorithm 2 shows the pseudocode of the updateVnext function, where the set $\mathcal{V}_{next}$ of next partially connected convoy candidates is updated with a new extended convoy candidate v_{new}. If there exists a convoy candidate v in $\mathcal{V}_{next}$ s.t. v is a sub-convoy of v_{new} by Definition 4 (lines 1-2), then v should be deleted and replaced by v_{new} in order to maintain a set of *maximal* convoy candidates in $\mathcal{V}_{next}$ as in lines 3-4. Otherwise, the new convoy candidate v_{new} can be safely added in the set $\mathcal{V}_{next}$ to be used in the next iteration (line 6).

B. Density-Connectivity Validation

The second phase of VCoDA takes as an input each partially connected convoy discovered in the previous section and obtains a set of truly valid convoys that additionally satisfy the density-connectivity among the group objects. We extend our PCCD algorithm to this end; scanning through each timestamp of a given partially connected convoy, if a current valid convoy candidate v is extended by either a *smaller* or a *lager* snapshot cluster c with sufficient common objects, the density-connectivity of the resulting extended valid convoy candidate v_{ext}=$v \cap c$ should be validated by *re-clustering* the moving objects in v_{ext} at every timestamp of v_{ext}.

Algorithm 3 DCVal(v_{pcc}=$\langle\{o_1,\ldots,o_m\},[t_a,t_b]\rangle$, ε, m, k)

```
1: V ← ∅; // set of current valid convoys
2: for each timestamp t of v_pcc do
3:    V_next ← ∅; // next set of valid convoys
4:    C ← DBSCAN(P_t(v_pcc), m, ε);
5:    initialize each cluster c∈C, convoy v∈V as in lines 6, 8 of Algo.1;
6:    for each pair of (v, c) ∈ V×C s.t. |v ∩ c| ≥ m do
7:       v_ext.validated ← false;  v_ext ← ⟨v∩c, [v.lifetime_start, t]⟩;
8:       if v = c then // immediate validation
9:          v_ext.validated←true; V_next←updateVnext(V_next, v_ext);
10:         v.extended ← true;  v.absorbed ← true;
11:         c.matched ← true;  c.absorbed ← true;
12:      else if v ⊂ c then // validation by re-clustering
13:         C_t ← DBSCAN(P_t(v_ext), m, ε);
14:         if C_t=∅ then  v_ext.validated←true; v_ext.lifetime_end←t-1;
15:         else if |C_t|=1 ∧ v_ext=C_t then // val. by single re-clustering
16:            v_ext.validated←true;V_next←updateVnext(V_next,v_ext);
17:            v.extended←true;v.absorbed←true;c.matched←true;
18:         else v_ext ← v_ext ∩ C_t;
19:      if not v_ext.validated then // recursive validation
20:         VC ← DCVal(v_ext, ε, m, k);
21:         for each validated convoy vc ∈ VC do
22:            v.extended ← true; c.matched ← true;
23:            if vc.lifetime_end = t then
24:               V_next ← updateVnext(V_next, vc);
25:            else
26:               if vc.lifetime ≥ k then V_val ← V_val ∪ {vc};
27:      if not v.extended or not v.absorbed then
28:         if v.lifetime ≥ k then V_val ← V_val ∪ {v};
29:   for each cluster c ∈ C s.t. not c.matched or not c.absorbed do
30:      V_next ← updateVnext(V_next, c);
31:   V ← V_next;
32: return  V_val ← V_val ∪ V;
```

Algorithm 3 shows the pseudocode of our density-connectivity validation of partially connected convoy (DCVal). As an input, it takes a partially connected convoy v_{pcc} (e.g., $\langle\{o_1,\ldots,o_m\},[t_a,t_b]\rangle$), a distance threshold ε, an integer m for the minimum number of objects, and an integer k for the minimum lifetime. The algorithm is similar to PCCD, except that the density-connectivity is verified for each extended valid convoy candidate v_{ext}=$v \cap c$ in lines 8-26. For each matching pair $v \in \mathcal{V}$ and $c \in \mathcal{C}$ with sufficient common objects (line 6), if their moving objects are the same (i.e., v=c), we know that all the moving objects in v form a dense group c by themselves at the consecutive timestamp. Therefore, the density-connectivity of v_{ext} is immediately validated without further re-clustering (lines 8-11). If a current valid convoy candidate v is extended by a *larger* snapshot cluster c (line 12), we perform a re-clustering of the moving objects in v_{ext} on their snapshot positions at the current timestamp (line 13), in order to confirm that all moving objects of v_{ext} are density-connected by themselves at the current timestamp. The extended valid convoy candidate v_{ext} is indeed valid only if the re-clustering results in a single cluster equivalent to v_{ext} (line 15), which is then added to the next valid convoy candidate set $\mathcal{V}_{next}$ to be used in the next iteration (line 16). If only a subset of moving objects in v_{ext} from a cluster, v_{ext} is replaced with this subset in line 18 so as to be subsequently validated. If the re-clustering does not result in such a single matching cluster ($|\mathcal{C}_t|$=1) or the current valid convoy candidate v is extended by a *smaller* snapshot cluster, the density-connectivity of the extended valid convoy candidate v_{ext} is now validated by a recursive call to DCVal in line 20, which obtains a set $\mathcal{VC}$ of valid convoy candidates from v_{ext}. Each valid convoy vc in $\mathcal{VC}$ is inserted to the set $\mathcal{V}_{next}$ to be further extended if its lifetime ends at the current timestamp (lines 23-24). Otherwise, it is returned to the set $\mathcal{V}_{val}$ of valid convoys as an output of validation as long as its lifetime is sufficient (line 26). As in PCCD, the current valid convoy candidates in $\mathcal{V}$ that are neither extended nor absorbed by any of clusters in C and have the lifetime of at least k are returned to the set $\mathcal{V}_{pcc}$ in lines 27-28. The current snapshot clusters in $\mathcal{C}$ that are neither matched nor absorbed by any of the current valid convoy candidates in $\mathcal{V}$ are added to the next valid convoy candidates $\mathcal{V}_{next}$ (lines 29-30), to be used in the next iteration. Note that the algorithm returns as an output a set $\mathcal{V}_{val}$ consisting of 1) truly valid convoys satisfying all five constraints in Definition 2 and 2) valid convoy candidates that may not satisfy the sufficient lifetime constraint due to the recursive usage. Hence, it requires a post-processing to remove such *short* valid convoy candidates from the final set of V_{val}.

Table II
VALIDATION PROCESS OF DCVAL

	Current/next valid convoys in $\mathcal{V}$	Validation
t_1	$v_1 \leftarrow c_1 \leftarrow \langle\{o_1,o_2,o_3,o_4,o_5\},[t_1,t_1]\rangle$	
t_2	$v_1 \leftarrow v_1 \cap c_2 \leftarrow \langle\{o_1,o_2,o_3,o_4,o_5\},[t_1,t_2]\rangle$	immediate
t_3	$v_1 \leftarrow$DCVal$(v_1 \cap c_3) \leftarrow \langle\{o_1,o_2,o_3\},[t_2,t_3]\rangle$	recursive
t_4	$v_1 \leftarrow v_1 \cap c_4 \leftarrow \langle\{o_1,o_2,o_3\},[t_2,t_4]\rangle$ $v_2 \leftarrow c_4 \leftarrow \langle\{o_1,o_2,o_3,o_4\},[t_4,t_4]\rangle$	re-clustering

Table II shows the validation process of DCVal, assuming that all five moving objects in Fig. 1(b) form a partially connected convoy $\langle\{o_1,o_2,\ldots,o_5\},[t_1,t_4]\rangle$ and using the parameters m=3 and k=2. At t_1, the cluster c_1 is set as a valid convoy candidate v_1. At t_2, since the current valid convoy candidate v_1 is extended by the cluster c_2 with the exactly same moving objects, the extended convoy is immediately validated without re-clustering. At t_3, since the current

convoy candidate v_1 is extended by a *smaller* cluster c_3, the extended valid convoy candidate $v_1 \cap c_3 = \langle \{o_1,o_2,o_3\}, [t_1,t_3] \rangle$ is validated by a recursive call to DCVal. As shown in Fig. 1(b), the objects $\{o_1,o_2,o_3\}$ do not form a cluster by themselves at t_1. As a result, a valid convoy candidate $\langle \{o_1,o_2,o_3\}, [t_2,t_3] \rangle$ with a *reduced* lifetime is returned and added to the next valid convoy candidate set to be used in the next iteration. At t_4, the current valid convoy candidate v_1 is extended by a *larger* cluster c_4. Hence, the density-connectivity of $\langle \{o_1,o_2,o_3\}, [t_3,t_4] \rangle$ is validated by re-clustering the snapshot positions of o_1,o_2,o_3 only at the current timestamp t_4.

IV. Experimental Evaluation

We evaluate the performance of our valid convoy discovery algorithms on three real-world moving object trajectory datasets. The performance of VCoDA is compared to that of an existing convoy discovery algorithm CMC (Coherent Moving Cluster) [7], [3]. All the algorithms were implemented in Matlab$^{\mathrm{TM}}$ and the experiments were run on a Pentium IV machine with 3.2GHz CPU and 2 GB of RAM, running on Windows Server 2003.

A. Dataset and Parameter Setting

Table III summarizes the information of the three datasets. The **hurricane** dataset[2] contains the Atlantic hurricanes' position in latitude and longitude sampled at 6-hourly intervals from the years 1950 through 2006. Each trajectory was preprocessed to have timestamps relative to its starting time to find more groups of hurricanes with similar development in space and time. The **truck** dataset[3] consists of 276 trajectories of 50 trucks delivering concrete to several construction places around Athens metropolitan area in Greece. The locations in latitude and longitude were sampled approximately every 30 seconds for 33 days [1]. In our experiment, the fleet of a truck object traced during a single day was considered as a distinct trajectory to find more valid convoys. The **commute** dataset[4] consists of 210 trajectories of two people tracing their daily commute in the Cook and Dupage counties of Illinois, USA. The locations in latitude and longitude were sampled every second by GPS systems for 6 month. Each commute trajectory during a single day was assumed to be obtained from a distinct moving object in our experiment. In all three datasets, the un-sampled positions at missing timestamps were linearly interpolated using the nearby measured location values to be aligned at a pre-defined set of timestamps.

Our valid convoy discovery algorithm require three input parameters m, k, and ε to discover all valid convoys, each consisting of at least m moving objects that are density-connected w.r.t. the density constraints m and ε during at least k consecutive timestamps. We assume that the parameters m and k determining the *smallest* and *shortest* convoys of interest in the dataset are given by the domain experts. In our experiments, the minimum number of objects m is commonly set to 3 for all two datasets and the minimum number of timestamps k (i.e., lifetime) is set to around 20-th quantile of the trajectory length. Given the number of minimum moving objects m=3, the reasonable range of distance threshold ε is estimated based on the density distribution of snapshot positions at each timestamp using the *sorted 3-dist graph* proposed in [8] for the hurricane dataset. For the truck and commute datasets where the fleet of vehicles are physically constrained on the road network, it is determined by the width of roads inspected from the visualization of trajectories on the spatial plane. The actual parameter values used in our experiments are summarized in Table III.

Table III
Settings for experiments

	Hurricane	Truck	Commute
Number of objects	608	276	210
Average trajectory length	31	407	1679
Total observations	18,951	112,203	354,913
Timestamp interval	6 hrs.	30 sec.	1 sec.
Number of timestamps	124	2875	48099
Number of objects (m)	3	3	3
Lifetime (k)	8	180	600
Distance threshold (ε)	1.0∼1.5	$\frac{1}{10^4} \sim \frac{6}{10^4}$	1∼5

B. Effectiveness

In order to compare our valid convoy discovery algorithms with CMC in terms of accuracy, we measure the precision and recall; let $\mathcal{V}_{val}$ be a result set of all valid convoys discovered by VCoDA and $\mathcal{V}_{cmc}$ be another set obtained by CMC. Considering $\mathcal{V}_{val}$ as a baseline, precision is defined as $\frac{|\mathcal{V}_{cmc} \cap \mathcal{V}_{val}|}{|\mathcal{V}_{cmc}|}$, indicating the probability that a convoy discovered by CMC is also retrieved by VCoDA. Precision attains a value 1 when the result set of CMC is entirely valid. Similarly, recall is defined as $\frac{|\mathcal{V}_{val} \cap \mathcal{V}_{cmc}|}{|\mathcal{V}_{val}|}$, implying the probability that a valid convoy discovered by VCoDA is also retrieved by CMC. Recall scores a value 1 when all valid convoys discovered by VCoDA are completely retrieved by CMC. Table IV shows the results of accuracy on all three datasets.

Over all three datasets, CMC generally found less number of convoys than VCoDA. This is expected because VCoDA additionally maintains unabsorbed snapshot clusters in the intermediate set of convoy candidates to be subsequently extended and output any unabsorbed convoy candidate on the fly while scanning the whole time span. Due to the same reason, CMC never discovered the complete set of valid convoys found by VCoDA, which is demonstrated in the low recall scores less than 1 and even closer to 0 in most of parameter settings. Also, only one third of the convoys

[2]http://weather.unisys.com/hurricane/atlantic/
[3]http://www.rtreeportal.org/
[4]http://cs.uic.edu/∼boxu/mp2p/gps_data.html

Table IV
COMPARISON OF ACCURACY

Hurricane: m=3, k=8 (48 hrs.)						
ε	1.0	1.1	1.2	1.3	1.4	1.5
$\lvert\mathcal{V}_{val}\rvert$	2	3	5	8	20	36
$\lvert\mathcal{V}_{cmc}\rvert$	3	5	7	17	18	15
precision	0.33	0.40	0.29	0.12	0.11	0.20
recall	0.50	0.67	0.40	0.25	0.10	0.08
Truck: m=3, k=180 (1hr. 30 min.)						
ε	0.0001	0.0002	0.0003	**0.0004**	0.0005	0.0006
$\lvert\mathcal{V}_{val}\rvert$	2	25	76	**145**	179	220
$\lvert\mathcal{V}_{cmc}\rvert$	1	12	14	**16**	18	18
precision	1.00	0.33	0.29	0.19	0.33	0.28
recall	0.50	0.16	0.05	0.02	0.03	0.02
Commute: m=3, k=600 (1min. 30 sec.)						
ε	1	2	3	4	5	6
$\lvert\mathcal{V}_{val}\rvert$	16	96	205	341	477	543
$\lvert\mathcal{V}_{cmc}\rvert$	7	16	20	23	25	25
precision	0.43	0.25	0.15	0.26	0.28	0.32
recall	0.19	0.04	0.01	0.02	0.01	0.01

discovered by CMC are valid on average as indicated by the low precision around 0.30, which is because of the missing density-connectivity validation process in CMC. In summary, VCoDA improves the precision by a factor of 3 on average and the recall by up to 2 orders of magnitude as compared to CMC.

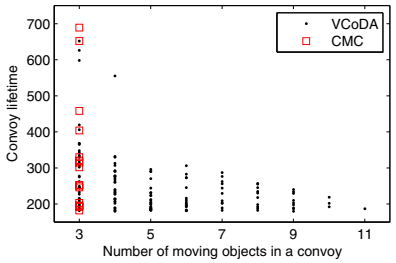


Figure 4. Comparison of discovered convoys

Figure 4 compares the size and the lifetime of 145 VCoDA and 16 CMC convoys found over the truck dataset with the distance threshold ε=0.0004 (see the bold-faced numbers in Table IV). In this figure, the X-axis represents the number of moving objects in a convoy and the Y-axis is the convoy lifetime (number of timestamps). As expected, CMC discovered only *small* convoys consisting of three moving objects and missed all *larger* convoys, which was commonly observed in other datasets. VCoDA discovers valid convoys represented in the dataset that the existing convoy discovery algorithm (CMC) is unable to find.

C. Processing Time

Figure 5 shows the processing time of VCoDA and CMC on three datasets, where the processing time of VCoDA is divided by a horizontal cut to denote the running time of PCCD (first phase for discovering partially connected convoys) and DCVal (second phase for density-connectivity validation) in the lower and upper bars, respectively. Over all three datasets, VCoDA took more time than CMC, which was expected due to the newly added density-connectivity validation of VCoDA. In general, the time complexity of both VCoDA (particularly PCCD) and CMC is affected by the total number of timestamps (T), the number of moving objects (N), the number of snapshot clusters ($|\mathcal{C}|$), and the number of intermediate convoy candidates ($|\mathcal{V}|$). While the first three factors are the same in both algorithms, VCoDA typically maintains much larger set $\mathcal{V}$ of convoy candidates to discover all partially connected convoys. Table V summarizes the maximum size of $\mathcal{V}$ encountered during the process of PCCD and CMC. The increased processing time of the first phase of VCoDA (i.e., PCCD) well explains the increased processing time of the first phase of VCoDA.

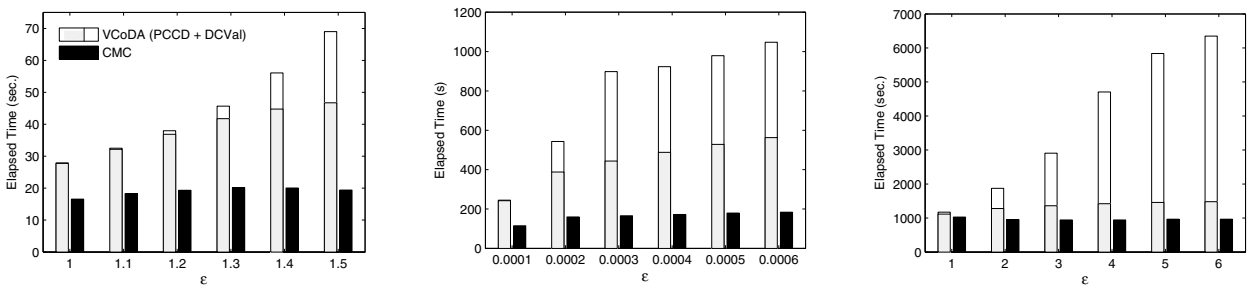
Table V
COMPARISON OF MAXIMUM $|\mathcal{V}|$

Hurricane: m=3, k=8 (48 hrs.)						
ε	1.0	1.1	1.2	1.3	1.4	1.5
PCCD	125	129	149	168	180	1835
CMC	68	71	77	74	71	62
Truck: m=3, k=180 (1hr. 30 min.)						
ε	0.0001	0.0002	0.0003	0.0004	0.0005	0.0006
PCCD	47	64	70	72	80	87
CMC	12	14	14	16	17	18
Commute: m=3, k=600 (1min. 30 sec.)						
ε	1	2	3	4	5	6
PCCD	14	21	26	27	28	28
CMC	7	8	6	4	3	3

The density-connectivity validation by DCVal took up to a factor of 2 less time than the partially connected convoy discovery by PCCD in the hurricane dataset (Figure 5(a)), where the average length of lifetime of partially connected convoys to be validated is 9.6 timestamps and the average number of moving objects is 4.4. In the truck dataset where the average lifetime of valid convoy candidates consisting of 5 moving objects on average is 261 timestamps, the running time of DCVal is comparable to that of PCCD. However, DCVal took relatively much more time up to a factor of 5 than PCCD in the commute dataset where we obtained valid convoy candidates consisting of 8 moving objects on average and with average lifetime of 1103 timestamps. These results might indicate that the overall running time of VCoDA is more dominated by the density-connectivity validation process for the partially connected convoys with *longer* lifetime.

V. RELATED WORK

We have already reviewed the most relevant work (moving cluster [4] and convoy [7], [3]) in Section I. Here, we just briefly discuss some of the other related work. Laube *et al.* [10] first introduced the notions of several relative motion patterns within groups of moving objects and proposed simple algorithms to discover these patterns. Here,



(a) Hurricane (b) Truck (c) Commute

Figure 5. Comparison of processing time in seconds

the term, *flock*, was first used to denote a cluster of snapshot locations of moving objects showing concurrence at a single instance of time. Gudmundsson *et al.* [6] developed efficient approximation algorithms to compute four spatio-temporal motion patterns: flock, leadership, convergence, and encounter. Later, Gudmundsson *et al.* extended the notion of *snapshot* flock to a long *duration* flock over a certain time interval. Several exact and approximation algorithms were proposed in [2] to discover longest duration flocks and the computation was improved by mapping spatio-temporal data into a high dimensional space and reducing the search of longest duration flocks into a sequence of standard range searches using a spatial indexing structure [11]. Wang *et al.* [5] introduced a user group pattern, which is defined as a group of users that are within a distance threshold from one another for at least a minimum duration. This group pattern generalizes the long duration flock in that the whole duration is not necessarily consecutive in time but composed with multiple time intervals. Although this group pattern relaxes the consecutive time constraint, it is still spatially limited to a group of fixed size and shape like flock. In addition, their algorithms are not directly applicable to our valid convoy discovery problem.

VI. Conclusions

In this paper, we showed that existing convoy discovery algorithms are inaccurate at finding valid convoys. Hence, we proposed a new algorithm VCoDA for the accuracy discovery of valid convoys form a set of moving object trajectories. In experiments on real datasets, VCoDA outperforms the current convoy algorithm in terms of precision and recall by a factor of 3 on average and up to 2 orders of magnitude, respectively. We intend to improve the efficiency of our VCoDA algorithm for future work.

Acknowledgment

This research has been funded in part by NSF grants IIS-0238560 (PECASE), IIS-0534761, IIS-0742811 and CNS-0831505 (CyberTrust), and in part from the METRANS Transportation Center, under grants from USDOT and Caltrans. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[1] F. Giannotti, M. Nanni, F. Pinelli, and D. Pedreschi, "Trajectory pattern mining," in *KDD '07*, 2007, pp. 330–339.

[2] J. Gudmundsson and M. van Kreveld, "Computing longest duration flocks in trajectory data," in *GIS'06*, 2006, pp. 35–42.

[3] H. Jeung, M. L. Yiu, X. Zhou, C. S. Jensen, and H. T. Shen, "Discovery of convoys in trajectory databases," *Proc. VLDB Endow.*, vol. 1, no. 1, pp. 1068–1080, 2008.

[4] P. Kalnis, N. Mamoulis, and S. Bakiras, "On discovering moving clusters in spatio-temporal data," in *Advances in Spatial and Temporal Databases*, 2005, pp. 364–381.

[5] Y. Wang, E.-P. Lim, and S.-Y. Hwang, "Efficient mining of group patterns from user movement data," *Data Knowl. Eng.*, vol. 57, no. 3, pp. 240–282, 2006.

[6] J. Gudmundsson, M. van Kreveld, and B. Speckmann, "Efficient detection of motion patterns in spatio-temporal data sets," in *GIS'04*, 2004, pp. 250–257.

[7] H. Jeung, H. T. Shen, and X. Zhou, "Convoy queries in spatio-temporal databases," in *ICDE'08*, April 2008, pp. 1457–1459.

[8] M. Ester, H. peter Kriegel, J. S, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *KDD'96*, 1996, pp. 226–231.

[9] H. Yoon and C. Shahabi, "Robust time-referenced segmentation of moving object trajectories," in *ICDM'08*, 2008, pp. 1121–1126.

[10] P. Laube and S. Imfeld, "Analyzing relative motion within groups of trackable moving point objects," in *GIScience'02*, 2002, pp. 132–144.

[11] M. Benkert, J. Gudmundsson, , F. Hubner, and T. Wolle, "Reporting flock patterns," in *14th European Symposium on Algorithms*, 2006, pp. 660–671.