Grouping time-varying data for interactive exploration

Citation for published version (APA):
Grouping Time-varying Data for Interactive Exploration

Arthur van Goethem† Marc van Kreveld§ Maarten Löffler† Bettina Speckmann* Frank Staals‡

1 Introduction
We present algorithms and data structures that support the interactive analysis of the grouping structure of one-, two-, or higher-dimensional time-varying data while varying all defining parameters. Grouping structures (which track the formation and dissolution of groups) characterize important patterns in the evolution of sets of time-varying data. We follow Buchin et al. [4] who define groups using three parameters: group-size, group-duration, and inter-entity distance.

Trajectory grouping structure [4]. Let \( \mathcal{X} \) be a set of \( n \) entities moving in \( \mathbb{R}^d \) and let \( T \) denote time. The entities trace trajectories in \( T \times \mathbb{R}^d \). We assume that each individual trajectory is piecewise linear and consists of at most \( \tau \) vertices. Two entities \( a \) and \( b \) are \( \varepsilon \)-connected at time \( t \) if there is a chain of entities \( a = c_1, c_2, \ldots, c_k = b \) such that for any pair of consecutive entities \( c_i \) and \( c_{i+1} \) the distance at time \( t \) is at most \( \varepsilon \). A set \( G \) is \( \varepsilon \)-connected, if for any pair \( a, b \in G \), the entities are \( \varepsilon \)-connected. Let \( \varepsilon \), and \( \delta \), a set of entities \( G \) is an \((m, \varepsilon, \delta)\)-group during time interval \( I \) if (and only if) (i) \( G \) has size at least \( m \), (ii) duration \((I) \geq \delta \), and (iii) \( G \) is \( \varepsilon \)-connected at any time \( t \in I \). An \((m, \varepsilon, \delta)\)-group \((G, I)\) is maximal if \( G \) is maximal in size or \( I \) is maximal in duration, that is, if there is no group \( H \supset G \) that is also \( \varepsilon \)-connected during \( I \), and no interval \( J \supset I \) such that \( G \) is \( \varepsilon \)-connected during \( J \).

Results and Organisation. We describe a data structure \( \mathcal{D} \) that represents the grouping structure, that is, its maximal groups, while allowing efficient change of the parameters. The complexity of the problem appears already in one-dimensional time-varying data. Hence we restrict our description to \( \mathbb{R}^1 \), the full paper extends our results to higher dimensions.

If all three parameters \( m \), \( \varepsilon \), and \( \delta \) can vary independently the question arises which constitutes a meaningful maximal group. Consider a maximal \((m, \varepsilon, \delta)\)-group \((G, I)\). If we slightly increase \( \varepsilon \) to \( \varepsilon' \), and consider a slightly longer time interval \( I' \supset I \) then \((G, I')\) is a maximal \((m, \varepsilon', \delta)\)-group. Intuitively, these groups \((G, I)\) and \((G, I')\) are the same. Thus, we are interested only in (maximal) groups that are “combinatorially different”. The set of entities \( G \) may also be a maximal \((m, \varepsilon, \delta)\)-group during a time interval \( J \) completely disjoint from \( I \), we also wish to consider \((G, I)\) and \((G, J)\) to be combinatorially different groups. In Section 2 we formally define when two (maximal) \((m, \varepsilon, \delta)\)-groups are (combinatorially) different. We prove that there are at most \( O(|A|n^2) \) such groups, where \( A \) is the arrangement of the trajectories in \( T \times \mathbb{R}^1 \), and \( |A| \) is its complexity. We also argue that the number of maximal groups may be as large as \( \Omega(n^3) \), even for fixed parameters \( m \), \( \varepsilon \), and \( \delta \) and in \( \mathbb{R}^1 \). This significantly strengthens the lower bound of Buchin et al. [4]. In Section 3 we present an \( O(|A|n^2 \log^2 n) \) time algorithm to compute all combinatorially different maximal groups.

In the full paper we describe a data structure that allows us to efficiently obtain all groups for a given set of parameter values. We also describe data structures for the interactive exploration of the data. Specifically, given the set of maximal \((m, \varepsilon, \delta)\)-groups we want to change one or more of the parameters and efficiently report only those maximal groups which either ceased to be a maximal group or became one. Our data structures can answer symmetric-difference queries [5].

2 Combinatorially Different Maximal Groups
We consider entities moving in \( \mathbb{R}^1 \), hence the trajectories form an arrangement \( A \) in \( T \times \mathbb{R}^1 \). Consider the four-dimensional parameter space \( P \) with axes time, size, distance, and duration. A set of entities \( G \) defines a region \( A_G \) in which it is alive: a point \((t, m, \varepsilon, \delta)\) lies in \( A_G \) if and only if \( G \) is an \((m, \varepsilon, \delta)\)-group at time \( t \). These regions help define when groups are combinatorially different. We start by fixing \( m = 1 \) and \( \delta = 0 \) to define and count the number of combinatorially different maximal \((1, \varepsilon, 0)\)-groups, over all choices of parameter \( \varepsilon \). Theorem 6 and Lemma 7 extend these results to include other values of \( \delta \) and \( m \).

Consider the \((t, \varepsilon)\)-plane in \( P \) through \( \delta = 0 \) and \( m = 1 \). The intersection of all regions \( A_G \) with this plane are the points \((t, \varepsilon)\) for which \( G \) is a \((1, \varepsilon, 0)\)-group. Note that \( G \) is a \((1, \varepsilon, 0)\)-group at time \( t \) if and only if the set \( G \) is \( \varepsilon \)-connected at time \( t \). \( A_G \), restricted to this plane, is simply connected. Furthermore, as the distance between any pair of entities moving in \( \mathbb{R}^1 \) varies linearly, \( A_G \) is bounded from below by a \( t \)-monotone polyline \( f_G \). The region is unbounded from above: if \( G \) is \( \varepsilon \)-connected (at time \( t \)) for some value \( \varepsilon \),
then it is also \( \varepsilon' \)-connected for any \( \varepsilon' \geq \varepsilon \) (see Fig. 1).

Every maximal length segment in the intersection between (the restricted) \( A_G \) and the horizontal line \( \ell_x \) at height \( \varepsilon \) corresponds to a (maximal) time interval \( I \) during which \( (G, I) \) is a \((1, \varepsilon, 0)\)-group, or an \( \varepsilon \)-group for short. Every such a segment corresponds to an instance of \( \varepsilon \)-group \( G \).

**Observation 1** Set \( G \) is a maximal \( \varepsilon \)-group on \( I \), iff the line segment \( s_{x,t} = \{(t, \varepsilon) \mid t \in I\} \) is a maximal length segment in \( A_G \), and is not contained in \( A_H \), for a supergroup \( H \supset G \).

Two instances of \( \varepsilon \)-group \( G \) may merge. Let \( v \) be a local maximum of \( f_G \) and \( I_1 = [t_1, v_1] \) and \( I_2 = [v_1, t_2] \) be two instances of group \( G \) meeting at \( v \). At \( v_2 \), the two instances \( G \) that are alive during \([t_1, v_1]\) and \([v_1, t_2]\) merge and we now have a single time interval \( I = [t_1, t_2] \) on which \( G \) is a group. We say that \( I \) is a new instance of \( G \) different from \( I_1 \) and \( I_2 \). We can thus decompose \( A_G \) into maximally-connected regions, each corresponding to a distinct instance of group \( G \), using horizontal segments through the local maxima of \( f_G \). We further split each region at the values \( \varepsilon \) where \( G \) changes between being maximal and being dominated. Let \( P_G \) denote the obtained set of regions in which \( G \) is maximal. Each such a region \( P \) corresponds to a *combinatorially distinct* instance on which \( G \) is a maximal group (with at least one member and duration at least zero). The region \( P \) is bounded by at most two horizontal line segments and two \( \varepsilon \)-monotone chains (see Fig. 1(b)).

**Counting maximal \( \varepsilon \)-groups.** To bound the number of distinct maximal \( \varepsilon \)-groups, over all values of \( \varepsilon \), we count the number of polygons in \( P_G \) over all sets \( G \). Consider a distinct instance (a set of entities \( G \) and a region \( P \in P_G \)) of the maximal \( \varepsilon \)-group \( G \). All vertices of \( P \) lie on the polyline \( f_G \): they are either vertices of \( f_G \), or they are points \( (t, \varepsilon) \) on the edges of \( f_G \) where \( G \) starts or stops being maximal. Any vertex is used by at most a constant number of regions from \( P_G \).

Below we show that the complexity of the arrangement \( \mathcal{H} \), of all polylines \( f_G \) over all \( G \), is bounded by \( O(|A|n) \). Furthermore, we show that each vertex of \( \mathcal{H} \) can be incident to at most \( O(n) \) regions. It follows that the complexity of all polygons \( P \in P_G \), over all groups (sets) \( G \), and thus also the number of such sets, is at most \( O(|A|n^3) \).

**The complexity of \( \mathcal{H} \).** The span \( S_G(t) = \{a \mid a \in \mathcal{X} \vee a(t) \in [\min_{b \in G} b(t), \max_{b \in G} b(t)]\} \) of a set of entities \( G \) at time \( t \) is the set of entities between the lowest and highest entity of \( G \) at time \( t \). Let \( h_a(t) \) denote the distance from entity \( a \) to the entity directly above \( a \) at time \( t \), that is, \( h_a(t) \) is the height of the face in \( \mathcal{A} \) that has \( a \) on its lower boundary at time \( t \).

**Observation 2** A set \( G \) is \( \varepsilon \)-connected at time \( t \), if and only if the largest nearest neighbor distance among the entities in \( S_G(t) \) is at most \( \varepsilon \). Hence

\[
f_G(t) = \max_{a \in S_G(t)} h_a(t).
\]

It follows that \( \mathcal{H} \) is actually the arrangement of the \( n \) functions \( h_a \), for \( a \in \mathcal{X} \). We use this fact to show that \( \mathcal{H} \) has complexity at most \( O(|A|n) \):

**Lemma 1** Let \( \mathcal{A} \) be an arrangement of \( n \) line segments, and let \( k \) be the maximum number of line segments intersected by a vertical line. The number of triplets \((F, F', x)\) such that the faces \( F \in \mathcal{A} \) and \( F' \in \mathcal{A} \) have equal height \( h \) at \( x \)-coordinate \( x \) is at most \( O(|A|k) \leq O(|A|n) \leq O(n^3) \).

**Lemma 2** The arrangement \( \mathcal{H} \) has size \( O(|A|n) \).

It remains to show that each vertex \( v \) of \( \mathcal{H} \) can be incident to at most \( O(n) \) polygons from different sets. Lemma 3 follows from Buchin et al. [4]:

**Lemma 3** Let \( R \) be the Reeb graph for a fixed value \( \varepsilon \) capturing the movement of a set of \( n \) entities moving along piecewise-linear trajectories in \( \mathbb{R}^3 \) (for some constant \( d \)), and let \( v \) be a vertex of \( R \). There are at most \( O(n) \) maximal groups that start or end at \( v \).

**Lemma 4** Let \( v \) be a vertex of \( \mathcal{H} \). Vertex \( v \) is incident to at most \( O(n) \) polygons from \( P = \bigcup_{G \in \mathcal{X}} P_G \).

---

Figure 1: (a) A set of trajectories for a set of entities moving in \( \mathbb{R}^3 \) (b) The region \( A_{(r,v)} \) during which \( \{r,v\} \) is alive, and its decomposition into polygons, each corresponding to a distinct instance. In all such regions, except the top one \( \{r,v\} \) is a maximal group: in the top region \( \{r,v\} \) is dominated by \( \{r,v,o\} \) (darker region).
Lemma 5 The number of distinct ε-groups, over all values ε, and the total complexity of all regions \( P = \bigcup_{G \subseteq X} P_G \), are both at most \( O(|H|n) = O(|A|n^2) \).

Theorem 6 Let \( X \) be a set of \( n \) entities, in which each entity travels along a piecewise-linear trajectory of \( \tau \) edges in \( \mathbb{R}^1 \), and let \( A \) be the resulting trajectory arrangement. The number of distinct maximal groups is at most \( O(|A|n^2) = O(\tau n^4) \), and the total complexity of all regions in the parameter space corresponding to these groups is also \( O(|A|n^2) = O(\tau n^4) \).

Lemma 7 For a set \( X \) of \( n \) entities, in which each entity travels along a piecewise-linear trajectory of \( \tau \) edges in \( \mathbb{R}^1 \), there can be \( \Omega(\tau n^3) \) maximal ε-groups.

3 Algorithm

We now refer to combinatorially different maximal groups simply as groups. Our algorithm computes a representation (of size \( O(|A|n^2) \)) of all groups, which we can use to list all groups and, given a pointer to a group \( G \), list all its members and the grouping polygon \( Q_G \in P_G \). We assume \( \delta = 0 \) and \( m = 1 \).

We use the arrangement \( H \) in the \((t, \varepsilon)\)-plane. Line segments in \( H \) correspond to the height function of the faces in \( A \). Let \( a, b \in S_G(t) \) be the pair of consecutive entities in the span of a group \( G \) with maximum vertical distance at time \( t \). The critical pair \((a, b)\) determines the minimal value of \( \varepsilon \) such that the group \( G \) is \( \varepsilon \)-connected at time \( t \). The distance between \((a, b)\) defines an edge of the polygon bounding \( G \) in \( H \).

Our representation consists of the arrangement \( H \) in which each edge \( e \) is annotated with a data structure \( T \), a list \( L \) with the top edge in each grouping polygon \( Q_G \in P_G \), and a data structure \( S \) to support reconstructing the grouping polygons.

We compute \( H \) in \( O(|H|) = O(\tau n^3) \) time \([1]\). Given \( H \) we use a sweep line algorithm to construct the representation. A horizontal line \( t(\varepsilon) \) is swept at height \( \varepsilon \) upwards, and all groups \( G \) whose grouping polygon \( Q_G \) currently intersects \( t \) are maintained. To achieve this we maintain a two-part status structure. First, a set \( S \) with for each group \( G \) the time interval \( I(G, \varepsilon) = Q_G \cap t(\varepsilon) \). We can implement \( S \) using any standard balanced binary search tree. Second, for each edge \( e \in H \) intersected by \( t(\varepsilon) \) a data structure \( T_e \) with the sets of entities whose time interval starts or ends at \( e \), that is, \( G \in T_e \) if and only if \( I(G, \varepsilon) = [s, t] \) with \( s = e \cap t(\varepsilon) \) or \( t = e \cap t(\varepsilon) \). The data structures \( T_e \) support the operations listed below.

In addition, we store with each interval \( I(G, \varepsilon) \) a pointer to the previous version of the interval \( I(G, \varepsilon') \) if (and only if) the starting time (ending time) of \( G \) changed to a different edge at \( \varepsilon' \).

The data structure \( T_e \). We need a data structure \( T = T_e \) that supports Filter, Insert, Delete, Merge, Contains, and HasSuperSet efficiently. We describe a structure of size \( O(n) \), that supports Contains and HasSuperSet in \( O(\log n) \) time, Filter in \( O(n) \) time, and Insert and Delete in amortized \( O(\log^2 n) \) time.

In general, answering Contains and HasSuperSet queries in a dynamic setting is hard and may require \( O(n^2) \) space \([6]\).

Lemma 8 Let \( G \) and \( H \) be two non-empty ε-groups that both end at time \( t \). We have:

\[
(G \cap H \neq \emptyset \land |G| \leq |H|) \iff G \subseteq H \land G \neq \emptyset.
\]

We implement \( T \) with a tree similar to the grouping-tree used by Buchin et al. \([4]\). Let \( \{G_1, \ldots, G_k\} \) denote the groups stored in \( T \), and let \( X' = \bigcup_{i \in [1, \ldots, k]} G_i \) denote the entities in these groups. Our tree \( T \) has a leaf
for every entity in $X'$. Each group $G_i$ is represented by an internal node $v_i$. For each internal node $v_i$ the set of leaves in the subtree rooted at $v_i$ corresponds exactly to the entities in $G_i$. By Lemma 8 these sets indeed form a tree. With each node $v_i$, we store the size of $G_i$, and an arbitrary entity in $G_i$. We preprocess $T$ in $O(n)$ time to support level-ancestor (LA) queries as well as lowest common ancestor (LCA) queries, using the methods of Bender and Farach-Colton [2, 3]. Both methods work only for static trees, whereas we need updates to $T$ as well. Since we query $T_e$ only when processing the upper end vertex of $e$, we can be lazy in updating $T_e$ and simply rebuild $T_e$ when needed.

**HasSuperSet and Contains queries.** Using LA queries we can do a binary search on the ancestors of a given node. This allows us to implement both $\text{HasSuperSet}(T_e, G)$ queries and $\text{Contains}(T_e, G)$ in $O(\log n)$ time for a group $G$ ending or starting on edge $e$. Let $a$ be an arbitrary element from group $G$. If the data structure $T_e$ contains a node matching the elements in $G$ then it must be an ancestor of the leaf containing $a$ in $T$. That is, it is the ancestor that has exactly $|G|$ elements. By Lemma 8 there is at most one such node. As ancestors get only more elements as we move up the tree, we find this node in $O(\log n)$ time by binary search. Similarly, we can implement the $\text{HasSuperSet}$ function in $O(\log n)$ time.

**Insert, Delete, and Merge queries.** The Insert, Delete, and Merge operations on $T_e$ are performed lazily; we execute them only when we get to the upper vertex of edge $e$. At such a time we may have to process a batch of $O(n)$ such operations which we can handle in $O(n \log^2 n)$ time.

**Lemma 9** Let $G_1, \ldots, G_m$ be maximal $\varepsilon$-groups, ordered by decreasing size, such that: (i) all groups end at time $t$, (ii) $G_i \supseteq G_{i+1}$, for all $i$, (iii) the largest group $G_1$ has size $s$, and (iv) the smallest group has size $|G_m| > s/2$. We then have that $G_i \supseteq G_{i+1}$ for all $i \in [1, \ldots, m-1]$.

**Lemma 10** Given two nodes $v_G \in T$ and $v_H \in T'$, representing the set $G$ respectively $H$, both ending at time $t$, we can test if $G \subseteq H$ in $O(1)$ time.

**Lemma 11** Given $m = O(n)$ nodes representing maximal $\varepsilon$-groups $G_1, \ldots, G_m$, possibly in different data structures $T_1, \ldots, T_m$, that all share ending time $t$, we can construct a new data structure $T$ representing $G_1, \ldots, G_m$ in $O(n \log^2 n)$ time.

The final function $\text{Filter}$ can easily be implemented in linear time by pruning the tree from the bottom up.

**Lemma 12** We can handle each event in $O(n \log^2 n)$ time.

**Reconstructing the grouping polygons.** Given a group $G$ we can construct the complete grouping polygon $Q_G$ in $O(|Q_G|)$ time, and list all group members of $G$ in $O(|G|)$ time. We have access to the top edge of $Q_G$. This is an interval $I(G, \varepsilon)$ in $S$, specifically, the version corresponding to $\varepsilon$, where $\varepsilon$ is the value at which $G$ dies as a maximal group. We then follow the pointers to the previous versions of $I(G, \varepsilon)$ to construct the left and right chains of $Q_G$. When we encounter the value $\varepsilon$ at which $G$ is born, these chains either meet at the same vertex, or we add the final bottom edge of $Q_G$ connecting them. To report the group members of $G$, we follow the pointer to $I(G, \varepsilon)$ in $S$. This interval stores a pointer to its starting edge $e$, and to a subtree in $T_e$ of which the leaves represent the entities in $G$.

**Analysis.** The list $L$ contains $O(g) = O(|A| n^2)$ entries (Theorem 6), each of constant size. The total size of all $S$’s is $O(|H| n)$: at each vertex of $H$, there are only a linear number of changes in the intervals in $S$. Each edge $e$ of $H$ stores a data structure $T_e$ of size $O(n)$. It follows that our representation uses a total of $O(|H| n) = O(|A| n^2)$ space. Handling each of the $O(|H|)$ nodes requires $O(n \log^2 n)$ time, so the total running time is $O(|A| n^2 \log^2 n)$.

**Theorem 13** Given a set $X$ of $n$ entities, in which each entity travels along a trajectory of $\tau$ edges, we can compute a representation of all $g = O(|A| n^2)$ combinatorial maximal groups $G$ such that for each group $G \in \mathcal{G}$ we can report its grouping polygon and its members in time linear in its complexity and size, respectively. The representation has size $O(|A| n^2)$ and takes $O(|A| n^2 \log^2 n)$ time to compute, where $|A| = O(\tau n^2)$ is the complexity of the trajectory arrangement.

**References**


