Group Movement Pattern Mining Algorithm for Data Compression

P.Saranya, P.Divya Rani, S.Dhivya Prabha, V.SathyaBama
M, Tech software Engineering, Periyar Maniammai University
Vallam, Tanjore -613403, India

Abstract

To reduce the data volume, various algorithms have been proposed for data compression and data aggregation. In object tracking applications, natural phenomena show that many creatures form large social groups and move in regular patterns. However, the previous works do not address application level semantics, such as the group relationships and movement patterns, in the location data. In this paper, we first introduce an efficient distributed mining algorithm to approach the moving object clustering problem and discover group movement patterns. Afterward, we propose a novel compression algorithm, based on the discovered group movement patterns to overcome the group data compression problem. Our experimental results show that the proposed compression algorithm effectively and efficiently reduces the amount of delivered data and reduces energy consumption expense for data transmission in WSNs.

Keywords--- Data compression, data aggregation, distributed mining, object tracking ,energy consumption.

I. INTRODUCTION

To reduce the data volume, various algorithms have been proposed for data compression and data aggregation. In object tracking applications, many natural phenomena show that objects often exhibit some degree of regularity in their movements. For example, the famous annual wildebeest migration demonstrates that the movements of creatures are temporally and spatially correlated. Biologists also have found that many creatures, such as elephants,zebra, whales, and birds, form large social groups when migrating to find food, or for breeding or wintering. A new challenge of finding moving animals belonging to the same group and identifying their aggregated group movement patterns. Therefore, under the assumption that objects with similar movement patterns are regarded as a group, we define the moving object clustering problem as given the movement trajectories of objects, partitioning the objects into nonoverlapped groups such that the number of groups is minimized. Then, group movement pattern discovery is to find the most representative movement patterns regarding each group of objects, which are further utilized to compress location data. Discovering the group movement patterns is more difficult than finding the patterns of a single object or all objects, because we need to jointly identify a group of objects and discover their aggregated group movement patterns. The constrained resource of WSNs should also be considered in approaching the moving object clustering problem. The temporal-and-spatial correlations in the movements of moving objects are modeled as sequential patterns in data mining to discover the frequent movement patterns. However, sequential patterns 1) consider the characteristics of all objects, 2) lack information about a frequent pattern's significance regarding individual trajectories, and 3) carry no time information between consecutive items, which make them unsuitable for location prediction and similarity comparison.

II. PROBLEM DESCRIPTION

We formulate the problem of this paper as exploring the group movement patterns to compress the location sequences of a group of moving objects for transmission efficiency. Consider a set of moving objects O = {o1, o2, . . . , on} and their associated location sequence data set S = {S_1, S_2, . . . , Sn}.

A. Definition 1.

Two objects are similar to each other if their movement patterns are similar. Given the similarity measure function sim_p[2] and a minimal threshold sim_{min}, o_i and o_j are similar if their similarity score $sim_p(o_i, o_j)$ is above sim_{min}, i.e., $sim_p(o_i, o_j) \geq sim_{min}$. The set of objects that are similar to oi is denoted by $so(o_i)$ ={ o_j|¥o_j € O, $sim_p(o_i, o_j) \geq sim_{min}$}.

B. Definition 2.

A set of objects is recognized as a group if they are highly similar to one another. Let g denote a set of objects. g is a group if every object in g is similar to at least a threshold of objects in g, i.e., ¥o_i €g, $\frac{|so(o_i)\cap g|}{|g|} \geq \gamma$
where γ is with default value (½)[3]

We formally define the moving object clustering problem as follows: Given a set of moving objects O together with their associated location sequence data set S and a minimal similarity threshold sim_{min}, the moving object clustering problem is to partition O into non overlapped groups, denoted by G ={g_1, g_2, . . . , g_i}, such that the number of groups is minimized, i.e., |G| is minimal. Thereafter, with the discovered group information and the obtained group movement patterns, the group data compression problem is to compress the location sequences of a group of moving objects for transmission efficiency. Specifically, we formulate the group data compression problem as a merge problem and an HIR problem. The merge problem is to combine multiple location sequences to reduce the overall sequence length, while the HIR problem targets to minimize the entropy of a sequence such that the amount of data is reduced with or without loss of information.

III. MINING OF GROUP MOVEMENT PATTERNS

To tackle the moving object clustering problem, we propose a distributed mining algorithm, which comprises the GMPMine and CE algorithms. First, the GMPMine algorithm uses a PST to generate an object's significant movement patterns and computes the similarity of two objects by using simp to derive the local grouping results. The merits of simp include its accuracy and efficiency: First, simp considers the significances of each movement pattern regarding to individual objects so that it achieves better accuracy in similarity comparison. For a PST can be used to predict a pattern's occurrence probability, which is viewed as the significance of the pattern regarding the PST, sim_p thus includes movement patterns' predicted occurrence probabilities to provide fine-grained similarity comparison.

Second, sim_p can offer seamless and efficient comparison for the applications with evolving and evolutionary similarity relationships. This is because sim_p can compare the similarity of two data streams only on the changed mature nodes of emission trees [36], instead of all nodes. To combine multiple local grouping results into a consensus, the CE algorithm utilizes the Jaccard similarity coefficient to measure the similarity between a pair of objects, and normalized mutual information (NMI) to derive the final ensembling result. It trades off the grouping quality against the computation cost by adjusting a partition parameter. In contrast to approaches that perform clustering among the entire trajectories, the distributed algorithm discovers the group relationships in a distributed manner on sensor nodes. As a result, we can discover group movement patterns to compress the location data in the areas where objects have explicit group relationships. Besides, the distributed design provides flexibility to take partial local grouping results into ensembling when the group relationships of moving objects in a specified subregion are interested. Also, it is especially suitable for heterogeneous tracking configurations, which helps reduce the tracking cost, e.g., instead of waking up all sensors at the same frequency, a fine-grained tracking interval is specified for partial terrain in the migration season to reduce the energy consumption. Rather than deploying the sensors in the same density, they are only highly concentrated in areas of interest to reduce deployment costs.

A. The Group Movement Pattern Mining (Gmpmine) Algorithm

To provide better discrimination accuracy, we propose a new similarity measure sim_p to compare the similarity of two objects. For each of their significant movement patterns, the new similarity measure considers not merely two probability distributions but also two weight factors, i.e., the significance of the pattern regarding to each PST. The similarity score sim_p of o_i and o_j based on their respective PSTs, T_i and T_j, is defined as follows:

$$sim_p(o_i,o_j) = -\log \frac{\sum_{s\in\tilde{S}}\sqrt{\sum_{\sigma\in\Sigma}\left(P^{T_i}(s\sigma)-P^{T_j}(s\sigma)\right)^2}}{2L_{max}+\sqrt{2}}, \quad (1)$$

where $\tilde{S}$ denotes the union of significant patterns (node strings) on the two trees. The similarity score sim_p includes the distance associated with a pattern s, defined as

$$\begin{aligned} d(s) &= \sqrt{\sum_{\sigma\in\Sigma}\left(P^{T_i}(s\sigma)-P^{T_j}(s\sigma)\right)^2} \\ &= \sqrt{\sum_{\sigma\in\Sigma}\left(P^{T_i}(s)\times P^{T_i}(\sigma|s)-P^{T_j}(s)\times P^{T_j}(\sigma|s)\right)^2}, \end{aligned}$$

where d(s) is the euclidean distance associated with a pattern s over T_i and T_j.

For a pattern s € T, P^T (s) is a significant value because the occurrence probability of s is higher than the minimal support P_{min}. If o_i and o_j share the pattern s, we have s € Ti and s € Tj, respectively, such that P^{Ti} (s) and P^{Tj} (s) are non negligible and meaningful in the similarity comparison. Because the conditional empirical probabilities are also parts of a pattern, we consider the conditional empirical probabilities P^T (σ|s) when calculating the distance between two PSTs. Therefore, we sum d(s) for all s € $\tilde{S}$ as the distance between two PSTs. Note that the distance between two PSTs is normalized by its maximal value, i.e., $2L_{max} + \sqrt{2}.$ We take the negative log of the distance between two PSTs as the similarity score such that a larger value of the similarity score implies a stronger similar relationship, and vice versa. With the definition of similarity score, two objects are similar to each other if their score is above a specified similarity threshold. The GMPMine algorithm includes four steps. First, we extract the movement patterns from the location sequences by learning a PST for each object. Second, our algorithm constructs an undirected, unweighted similarity graph where similar objects share an edge between each other. We model the density of group relationship by the connectivity of a subgraph, which is also defined as the minimal cut of the subgraph. When the ratio of the connectivity to the size of the subgraph is higher than a threshold, the objects corresponding to the subgraph are identified as a group. Since the optimization of the graph partition problem is intractable in general, we bisect the similarity graph in the following way. We leverage the HCS cluster algorithm to partition the graph and derive the location group information. Finally, we select a group PST T_g for each group in order to conserve the memory space by using the formula expressed as T_g= argmax $_{T€T} \sum_{s€ \bar{S}} P^T$ (s), where S̄ denotes sequences of a group of objects and T̄ denotes their PSTs.

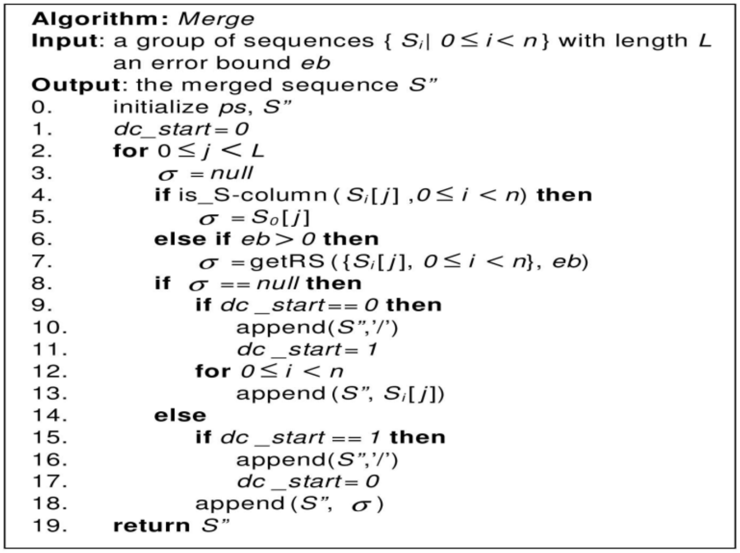
B. *The Cluster Ensembling (CE) Algorithm*

In the previous section, each CH collects location data and generates local group results with the proposed GMPMine algorithm. Since the objects may pass through only partial sensor clusters and have different movement patterns in different clusters, the local grouping results may be inconsistent. For example, if objects in a sensor cluster walk close together across a canyon, it is reasonable to consider them a group. In contrast, objects scattered in grassland may not be identified as a group. Furthermore, in the case where a group of objects moves across the margin of a sensor cluster, it is difficult to find their group relationships. Therefore, we propose the CE algorithm to combine multiple local grouping results. The algorithm solves the inconsistency problem and improves the grouping quality. The ensembling problem involves finding the partition of all moving objects O that contains the most information about the local grouping results. The algorithm includes three steps. First, we utilize Jaccard Similarity Coefficient as the measure of the similarity for each pair of objects. Second, for each δ € D, we construct a graph where two objects share an edge if their Jaccard Similarity Coefficient is above δ. Our algorithm partitions the objects to generate a partitioning result $G_δ$. Third, we select the ensembling result $G_δ$'.

IV. Compression Algorithm With Group Movement Patterns

Transmission of data is one of the most energy expensive tasks in WSNs, data compression is utilized to reduce the amount of delivered data. Therefore, to reduce the amount of delivered data, we propose the 2P2D algorithm. The algorithm includes the sequence merge phase and the entropy reduction phase to compress location sequences vertically and horizontally. In the sequence merge phase, we propose the Merge algorithm to compress the location sequences of a group of moving objects. Since objects with similar movement patterns are identified as a group, their location sequences are similar. The Merge algorithm avoids redundant sending of their locations, and thus, reduces the overall sequence length. It combines the sequences of a group of moving objects by 1) trimming Fig. 4. Design of the two-phase and 2D compression algorithm. multiple identical symbols at the same time interval into a single symbol or 2) choosing a qualified symbol to represent them when a tolerance of loss of accuracy is specified by the application. Therefore, the algorithm trims and prunes more items when the group size is larger and the group relationships are more distinct. Besides, in the case that only the location center of a group of objects is of interest, our approach can find the aggregated value in the phase, instead of transmitting all location sequences back to the sink for postprocessing. In the entropy reduction phase, we propose the Replace algorithm that utilizes the group movement patterns as the prediction model to further compress the merged sequence.

The Replace algorithm guarantees the reduction of a sequence’s entropy, and consequently, improves compressibility without loss of information. Specifically, we define a new problem of minimizing the entropy of a sequence as the HIR problem. we prove that the Replace algorithm obtains the optimal solution of the HIR problem as Theorem 1. in this section, we concentrate on the problem of compressing multiple similar sequences of a group of moving objects.



```
Algorithm: Merge
Input: a group of sequences { $S_i$| $0 \le i < n$} with length $L$
       an error bound $eb$
Output: the merged sequence $S''$
0.    initialize $ps$, $S''$
1.    $dc\_start = 0$
2.    for $0 \le j < L$
3.        $\sigma = null$
4.        if is_S-column ( $S_i[j]$ , $0 \le i < n$) then
5.            $\sigma = S_0[j]$
6.        else if $eb > 0$ then
7.            $\sigma$ = getRS ({$S_i[j]$, $0 \le i < n$}, $eb$)
8.        if $\sigma == null$ then
9.            if $dc\_start == 0$ then
10.               append($S''$,'/')
11.               $dc\_start = 1$
12.           for $0 \le i < n$
13.               append ($S''$, $S_i[j]$)
14.       else
15.           if $dc\_start == 1$ then
16.               append($S''$,'/')
17.               $dc\_start = 0$
18.           append ($S''$, $\sigma$)
19.   return $S''$
```

Fig 1.The merge algorithm

To compress the location sequences for a group of moving objects, we propose the Merge algorithm shown in Fig1. The input of the algorithm contains a group of sequences $\{S_i|0 \le i < n\}$ and an error bound eb, while the output is a merged sequence that represents the group of sequences. Specifically, the Merge algorithm processes the sequences in a column wise way. For each column, the algorithm first checks whether it is an S-column. For an S-column,

it retains the value of the items as Lines 4-5. Otherwise, while an error bound eb > 0 is specified, a representative symbol is selected according to the selection criterion as Line 7. If a qualified symbol exists to represent the column, the algorithm outputs it as Lines 15-18. Otherwise, the items in the column are retained and wrapped by a pair of “/” as Lines 9-13. The process repeats until all columns are examined. Afterward, the merged sequence S'' is generated.

A. Entropy Reduction Phase

In the entropy reduction phase, we propose the Replace algorithm to minimize the entropy of the merged sequence obtained in the sequence merge phase.

Definition 3 (HIR problem):

Given a sequence $S = \{s_i|s_i$ € $\sum, 0 \le i < L\}$ and a taglst, an intermediate sequence is a generation of S, denoted by $S' = \{s_i'| 0 \le i < L\}$, where s_i' is equal to s_i if taglst[i]=0. Otherwise, s_i’ is equal to s_i or ‘.’. we derive the first replacement rule—the accumulation rule: Replace all items of symbol σ in $\hat{S}'$where $n(\sigma) = n_{hit}(\sigma)$

Three Derived Replacement Rules:

1)The HIR problem is to find the intermediate sequence S' such that the entropy of S' is minimal for all possible intermediate sequences.

2)we derive the second replacement rule—the concentration rule: Replace all predictable items of symbol σ in $\hat{S}'$, where $n(\sigma) \le n(‘.’)$ or $n_{hit}(\sigma) > n(\sigma)=n(‘.’)$.

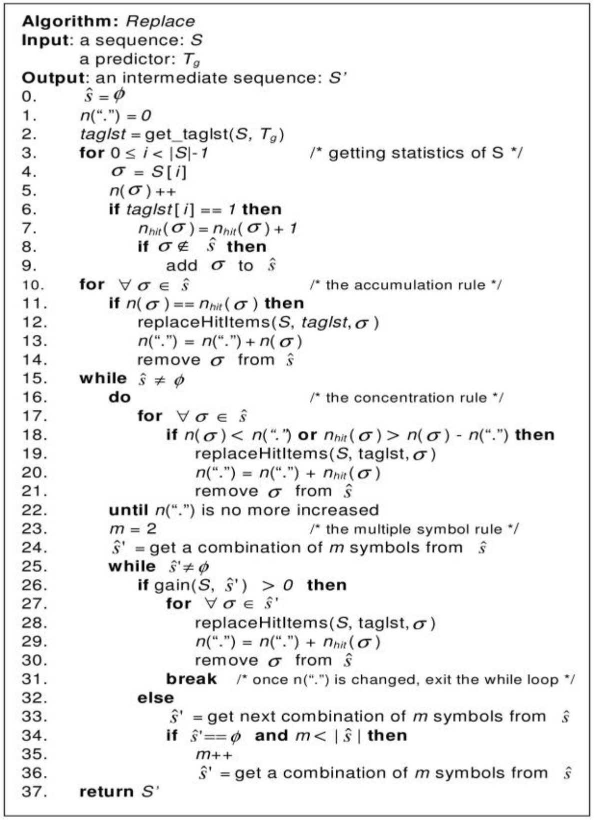
3)we derive the third replacement rule—the multiple symbol rule: Replace all of the predictable items of every symbol in

$\hat{S}'$ if $gain(\hat{S}') > 0$.

To solve the HIR problem, we explore properties of Shannon’s entropy to derive three replacement rules that our Replace algorithm leverages to obtain the optimal solution.

B. The Replace Algorithm:

Based on the observations described in the previous section, we propose the Replace algorithm that leverages the three replacement rules to obtain the optimal solution for the HIR problem.



```
Algorithm: Replace
Input: a sequence: $S$
        a predictor: $T_g$
Output: an intermediate sequence: $S'$
        $\hat{s} = \phi$
        $n(“.”) = 0$
        $taglst$ = get_taglst($S$, $T_g$)
        for $0 \leq i < |S|-1$                    /* getting statistics of S */
            $\sigma = S[i]$
            $n(\sigma)$++
            if $taglst[i] == 1$ then
                $n_{hit}(\sigma) = n_{hit}(\sigma) + 1$
                if $\sigma \notin \hat{s}$ then
                    add $\sigma$ to $\hat{s}$
        for $\forall \sigma \in \hat{s}$                    /* the accumulation rule */
            if $n(\sigma) == n_{hit}(\sigma)$ then
                replaceHitItems($S$, $taglst$, $\sigma$)
                $n(“.”) = n(“.”) + n(\sigma)$
                remove $\sigma$ from $\hat{s}$
        while $\hat{s} \neq \phi$
            do                                  /* the concentration rule */
                for $\forall \sigma \in \hat{s}$
                    if $n(\sigma) < n(“.”)$ or $n_{hit}(\sigma) > n(\sigma) - n(“.”)$ then
                        replaceHitItems($S$, taglst, $\sigma$)
                        $n(“.”) = n(“.”) + n_{hit}(\sigma)$
                        remove $\sigma$ from $\hat{s}$
            until $n(“.”)$ is no more increased
            $m = 2$                              /* the multiple symbol rule */
            $\hat{s}'$ = get a combination of $m$ symbols from $\hat{s}$
            while $\hat{s}' \neq \phi$
                if gain($S$, $\hat{s}'$) $> 0$ then
                    for $\forall \sigma \in \hat{s}'$
                        replaceHitItems($S$, taglst, $\sigma$)
                        $n(“.”) = n(“.”) + n_{hit}(\sigma)$
                        remove $\sigma$ from $\hat{s}$
                    break  /* once n(“.”) is changed, exit the while loop */
                else
                    $\hat{s}'$ = get next combination of $m$ symbols from $\hat{s}$
                    if $\hat{s}' == \phi$ and $m < |\hat{s}|$ then
                        $m$++
                         $\hat{s}'$ = get a combination of $m$ symbols from $\hat{s}$
        return $S'$
```

Fig 2. The Replace Algorithm

V. CONCLUSIONS

In this work, we exploit the characteristics of group movements to discover the information about groups of moving objects in tracking applications. We propose a distributed mining algorithm, which consists of a local GMPMine algorithm and a CE algorithm, to discover group movement patterns. With the discovered information, we devise the 2P2D algorithm, which comprises a sequence merge phase and an entropy reduction phase. In the sequence merge phase, we propose the Merge algorithm to merge the location sequences of a group of moving objects with the goal of reducing the overall sequence length. In the entropy reduction phase, we formulate the HIR problem and propose a Replace algorithm to tackle the HIR problem. In addition, we devise and prove three replacement rules, with which the Replace algorithm obtains the optimal solution of HIR efficiently. Our experimental results show that the proposed compression algorithm effectively reduces the amount of delivered data and enhances compressibility and, by extension, reduces the energy consumption expense for data transmission in WSNs.

References

1. S.S. Pradhan, J. Kusuma, and K. Ramchandran, *“Distributed Compression in a Dense Microsensor Network,”* IEEE Signal Processing Magazine, vol. 19, no. 2, pp. 51-60, Mar. 2002.
2. A. Scaglione and S.D. Servetto, *“On the Interdependence of Routing and Data Compression in Multi-Hop Sensor Networks,”* Proc. Eighth Ann. Int’l Conf. Mobile Computing and Networking, pp. 140-147, 2002.
3. N. Meratnia and R.A. de By, “A New Perspective on Trajectory Compression Techniques,” Proc. ISPRS Commission II and IV, WG II/5, II/6, IV/1 and IV/2 Joint Workshop Spatial, Temporal and Multi- Dimensional Data Modelling and Analysis, Oct. 2003.
4. S. Baek, G. de Veciana, and X. Su, “Minimizing Energy Consumption in Large-Scale Sensor Networks through Distributed Data Compression and Hierarchical Aggregation,” IEEE J. Selected Areas in Comm., vol. 22, no. 6, pp. 1130-1140, Aug. 2004.
5. C.M. Sadler and M. Martonosi, “Data Compression Algorithms for Energy-Constrained Devices in Delay Tolerant Networks,” Proc. ACM Conf. Embedded Networked Sensor Systems, Nov. 2006.
6. Y. Xu and W.-C. Lee, “Compressing Moving Object Trajectory in Wireless Sensor Networks,” Int’l J. Distributed Sensor Networks, vol. 3, no. 2, pp. 151-174, Apr. 2007.