A FAST INCREMENTAL CLUSTERING ALGORITHM BASED ON GRID AND DENSITY

ZHOU CHEN¹, XIANG-SHUANG LIU²

1. Lab of Computer Application, Qingdao University of Science and Technology, Qingdao 266061, China
2. Hisense Mobile Technology Ltd., Qingdao 266071, China
E-MIAL: xinqingcz78@hotmail.com

Abstract:
In recent years with data becoming larger and larger, the clustering algorithm not only needs to have high executing efficiency, but also is required to discover clusters with arbitrary shape and be insensitive to noise data. In this paper, a new kind of incremental clustering algorithm called ICECPG (incremental clustering using extended condensation point and grid) is presented in order to realize the real time clustering of the dynamic data. The innovations of these algorithms are capturing the shape of data space by extended condensation points, and then using grid-based and density-based clustering methods based on the theories of climbing hill algorithm and connectedness to cluster the data, guided by the difference data to implement incremental cluster. These algorithms have the ability of grid-based and density-based clustering methods’ good features, overcoming the traditional grid-based clustering method’s shortcoming of clustering quality debasement resulted by little or no consideration of data distribution when partitioning the grids. They can also decrease the number of region query and calculate in traditional density-based clustering method, which consequently reduces the I/O cost. So they have linear time complexity, and can be used in mining dynamic large databases.

Keywords:
Incremental clustering; grid; extended condensation point;

1. Introduction

Clustering is a descriptive task which groups a set of data without a predefined class attribute to maximize the intra-class similarity and minimize the interclass similarity. It is an important data-mining technique used to discover potential information distribution and knowledge from a great deal of information [1, 7, 8, 12]. Presently several frequent clustering algorithms include: splitting method [2] (K-means), hierarchy method [3] (CURE), density-based method [4, 11] (DBSCAN), grid-based method [5, 13] (CLIQUE) and model-based method [6] (COBWEB). However these algorithms mentioned above are usually just available for the clustering of static data, the clustering results may become old after a series of updates to the databases, which may lead to mistakes in decision support. So it is important to keep the clustering results up to date. Due to the large size of the databases and the high time complexity of clustering algorithms, it is highly desirable to perform these updates incrementally.

In this paper we first introduce a clustering algorithm- CGDCP [14] by calculating the condensation points of the data, then partitioning the condensation point space into a number of units and dealing with units instead of points. The innovations of CGDCP are capturing the shape of data space by condensation points, and then using grid-based and density-based clustering methods based on the theories of climbing hill algorithm [9, 10] and connectedness to cluster the data. After that we introduce an incremental clustering algorithm called ICECPG. The innovations of ICECPG are recalculating the extended condensation point guided by the difference data, and then using theories of climbing hill algorithm and connectedness to re-cluster the units where the difference data locate.

The rest of the paper is organized as follows: Section 2 presents a clustering algorithm-CGDCP followed by some definitions. Section 3 introduces a fast incremental clustering algorithm-ICECPG based on CGDCP. Section 4 analyses the performance of these two algorithms. Lastly, we conclude the paper.

2. CGDCP Clustering Algorithm

Let \( A = \{A_1, A_2, \cdots, A_d\} \) be a set of numeric attributes (dimensions), each dimension having a bounded, totally ordered domain. \( S = A_1 \times A_2 \times \cdots \times A_d \) is the minimum bounding hyper-rectangle of the database, constructing a d-dimensional numeric space. \( V = \{v_1, v_2, \cdots, v_n\} \) is a set of d-dimensional points, where \( v_i = \{v_{i1}, v_{i2}, \cdots, v_{id}\} \). By input parameters \( \xi_1, \xi_2, \cdots, \xi_d \), we define any rectangle units...
\( s_i \ (s_i \subset S) \) as a unit \( U_i = \{ u_{i1}, u_{i2}, \ldots, u_{id} \} \), where \( u_{ij} = (l_{ij}, l_{ij} + \xi_j) \). A d-dimensional point \( v_i = \{ v_{i1}, v_{i2}, \ldots, v_{id} \} \) is contained in \( U_i \), if and only if \( l_{ij} \leq v_{ij} < l_{ij} + \xi_j \), \( 1 \leq j \leq d \).

The density of \( U_i \) is defined as the number of points contained in it, denoted as \( D(U_i) \). A unit \( U_i \) is called dense if \( D(U_i) > \tau \), where \( \tau \) is a density threshold.

**Definition 1.** The geometrical centers of dense units are defined as condensation points. **Definition 2.** The points and condensation points in the dense units are defined as extended condensation points. CP is a density threshold.

We partition the data space \( S \) into non-overlapping rectangle grids, with the \( j^{th} \) dimension being divided into \( m_j \) intervals of equal length. Given an order to the intervals of each dimension, \( \{ g_1, g_2, \ldots, g_d \} \) is called the position of a grid. Assuming \( m = m_1 \) for all dimensions, \( S \) is divided into \( m_d \) grids. Let \( l_j \) be the length of the intervals of the \( j^{th} \) dimension. A d-dimensional extended condensation point \( v_i = \{ v_{i1}, v_{i2}, \ldots, v_{id} \} \) is contained in \( G \) having the position \( \{ g_1, g_2, \ldots, g_d \} \), if and only if \( (r_j - 1) \times l_j \leq v_{ij} < r_j \times l_j \), \( 1 \leq j \leq d \). The density of \( G \) is defined as the number of extended condensation points contained in it, denoted as \( D(G) \). A grid \( G \) is called non-empty if \( D(G) > 0 \).

**Definition 3.** We can define two relations between two grids \( G_i \) and \( G_j \) as follows:

1. \( G_i \) is a neighbor of \( G_j \) if and only if \( l_{ij} = j_i, m \in [1, d] \) or \( l_{ij} = j_i, l \neq m \) (1) or \( l_{ij} = j_i, l = m, j_i = m \) (2).

2. \( G_i \) is reachable from \( G_j \) if and only if: (1) there is another grid \( G_k \) such that both \( G_i \) and \( G_j \) are neighbors of \( G_k \); (2) there are grids \( G_i \) and \( G_j \) such that \( G_i \) is a neighbor of \( G_k \), \( G_j \) is a neighbor of \( G_i \), and \( G_k \) is reachable from \( G_j \).

We can summarize two properties from the above definition:

**Property 1.** The neighborhood relationship satisfies commutative law: \( G_i \) is a neighbor of \( G_j \) if and only if \( G_j \) is a neighbor of \( G_i \).

**Property 2.** The reachable relationship satisfies commutative law: \( G_i \) is reachable from \( G_j \) if and only if \( G_j \) is reachable from \( G_i \).

**Definition 4.** Connecting relationship is defined as either neighborhood relationship or reachable relationship.

**Property 3.** \( G_i \) connects with \( G_j \) if and only if there is a chain of grids \( G_i, G_k, \ldots, G_j \) in which every adjoining grids are neighbors to each other.

**Definition 5.** The grid \( G_i \), the density of which is not less than any density of grid \( G_j \) directly reachable from \( G_i \), is called local optimum grid.

There is a theorem of local optimum grid summarized from the definition 4:

**Property 4.** If \( G_i \) is directly reachable from \( G_j \) and both of them are local optimum grid, then \( D(G_j) = D(G_i) \).

**Proof.** Assuming this theorem is not tenable. So \( G_i \) is directly reachable from \( G_j \); \( G_i \) and \( G_j \) are local optimum grid, and \( D(G_i) > D(G_j) \). It is paradox with the content defined by definition 4 that the density of local optimum grid is no less than anyone of its directly reachable. Therefore the hypothesis is not tenable.

**Definition 6.** Clustering is defined as a set of grids, which are composed of the connecting local optimum grids and all the non-local optimum grids reachable from these connecting local optimum ones.

In CGDCP algorithm, the clustering process is to find a chain from every non-empty grid to the local optimum grids by heuristic search method, and unite the connecting local optimum grids to form clusters.

2.1. Algorithm description

In general, CGDCP can be divided into four steps:

**Step 1.** Find extended condensation points: First calculate the density of every unit in the data space; secondly calculate the condensation points of dense units; thirdly map each extended condensation points into the corresponding grid.

**Step 2.** Calculate the density of each non-empty grid and store its density, position and initialize its extended position.

Begin

For each grid \( G_i \):

\[
\text{if } D(G_i) > 0 \\
\{G_i\text{.density} = D(G_i); \}
\]

\[
G_i \text{.position} = \text{current position}; \]

\[
G_i \text{.extended positon} = G_i \text{.position}; \]

End
3. ICECPG Clustering Algorithm

Assuming we have got a set of clusters \( \mathcal{C} \) using the CGDCP algorithm. When new data are inserted or original data are deleted, we must modify the existed clusters to reflect the changes. Let \( \nabla V \) be the deleted data from \( V \), and \( \Delta V \) be the inserted data to \( V \). The new database \( V' = V \cup \Delta V \setminus \nabla V \). Since updates can be seen as a series of insertions and deletions, we only consider insertions and deletions. So we define the difference-data \( V_d \) as the set of \( \Delta V \cup \nabla V \). A unit is called difference-unit \( U_d \) if at least one difference-data is located in it. Since an inserted or deleted point only affects the density of unit in which it is contained, we consider the \( \nabla U_d \) which is dense before updating but not dense after updating and \( \Delta U_d \) which is not dense before updating but dense after updating.

After some insertions, non-dense units may become dense that extended condensation points increase, denoted by \( \Delta CP \). After some deletions, dense units may become non-dense that extended condensation points decrease, denoted by \( \nabla CP \). We define the difference-points as \( \Delta CP \cup \nabla CP \). When we confirm the difference-points, the difference-grids \( G_d \), which contain the difference-points, are confirmed.

3.1. Algorithm description

In general, ICECPG can be divided into five steps:

Step 1. Discover difference-data:

Begin
\[
\begin{align*}
\Delta V &= \emptyset \\
\nabla V &= \emptyset \\
\text{For each updating data } V_i \\
\text{If } (V_i \in V) \land (V_i \not\in V') \\
\text{Then } \nabla V &= \nabla V \cup \{V_i\} \\
\text{If } (V_i \in V) \land (V_i \not\in V') \\
\Delta V &= \Delta V \cup \{V_i\}
\end{align*}
\]
End

Step 2. Discover difference-unit and difference-point:

Begin
\[
\begin{align*}
\text{For each unit } U_i \text{ having } V_d \\
\text{Calculate } D(U_i) \text{ of the original data} \\
\text{If } (D(U_i) > \tau) \\
\{\text{Delete all the CPs } \in U_i \text{ from } \mathcal{C} \}
\end{align*}
\]
For each unit \( U_i \) having \( V_d \)
\[
\begin{align*}
\text{Calculate } D(U_i) \text{ of the updated data} \\
\text{If } (D(U_i) > \tau)
\end{align*}
\]

CGDCP algorithm.

2.2. Time complexity

DBSCAN computes the neighbor points for every point, since it does not know whether a point is a core point or a boundary point without searching its neighborhood. In CGDCP, we compute the density of every unit in step one. If the unit is dense, the condensation point is calculated and the extended condensation points are confirmed. Then we partition the extended condensation points into grids and deal with grids instead of points during step two. Only those non-empty grids are searched heuristically during step three. And those connecting sub-clusters are united in step four. In fact, the number of sub-cluster \(|\mathcal{C}|\) is much less than the grids \(|G|\) in the data space. So the time complexity of CGDCP is \( O(|U|+(1+n)|G|) \), where \( n \) is the average number of heuristic search steps.

CGDCP has the following properties:

1) CGDCP only stores the non-empty grids, the number of which is linear to \( N \) as the dimensionality increases so that CGDCP is applicable for high dimensionality;
2) CGDCP only search directly reachable and reachable grids, the number of which is much smaller than that of the non-empty grids;
3) If a sub-cluster is not connecting to the nearest sub-cluster, then it will form a cluster solely.

All the above properties improve the efficiency of the
Add all the \( CP_j \in U_i \) to \( \{ C \} \)}

End

Step 3. Partition difference-points into difference-grids and calculate their density

Step 4. Adopt heuristic search method to deal with every difference-grids:

Begin

\( \text{Flag}=0; \)

do 

For each difference-grid \( G_d \)

\{For each grid \( G_j \) directly reachable from \( G_d \)

\{If \( G_d \) is not the local optimum grid

\( G_d \).position = \( G_j \).position;\)

update all the grids positions of which are \( G_d \) to \( G_j \)

\( \text{Flag}=1 \} \}

\} While (\( \text{Flag}=1 \})

All the grids having the same position form a sub-cluster; store the position of each sub-cluster.

End

Step 5. Unite connecting sub-clusters \( C_i \); All the sub-clusters reachable from each other form a cluster.

3.2. Time complexity

In ICECPG, we search difference-data in step one; compute the density of every difference-unit and discover difference-points in step two. If a unit does not contain difference-data, the ICECPG algorithm will not deal with it in step two. In step three and step four, only the difference-points and difference-grid are considered instead of all the extended condensation points and grids. In step five all the sub-clusters are searched to determine the connecting ones. So the time complexity of ICECPG is \( O(|V_d|+|U_d|+(1+n)|G_d|) \), where \( n \) is the average number of heuristic search steps.

ICECPG has the following properties:

1) ICECPG only stores the difference-points and difference-grids, the number of which is linear to \( N \) as the dimensionality increases so that ICECPG is applicable for high dimensionality;

2) ICECPG only searches the difference-data, difference-units, difference-points and difference-grids instead of all the data, units, extended condensation points and grids. In fact, the number of difference-data \( |V_d| \), difference-unit \( |U_d| \), difference-grids \( |G_d| \) and difference-points \( |C_i| \) is respectively much less than the number of all the data \( |V| \), units \( |U| \), grids \( |G| \) and extended condensation points \( |CP| \) in the data space.

Known by all the above properties, the ICECPG algorithm is more efficient than the CGDCP algorithm. So ICECPG is more suitable for the data mining of large dynamic database.

4. Experimental Results

There are two evaluation criteria used to weigh the result of clustering:

(1) Rate of contractility (\( C_r \)):

\[ C_r = \left( \frac{m_{best}}{m_{result}} \right) \times 100\% \]

(The denominator denotes the clustering number of execution result; the numerator denotes the number of perfect clustering.)

(2) Rate of veracity (\( V_r \)):

\[ V_r = \left( \frac{m_{right}}{m_{all}} \right) \times 100\% \]

(The denominator denotes the number of all the data; the numerator denotes the number of correct-clustering data.)

In order to verify their availability, the CGDCP and ICECPG algorithms have been used in web text mining. The original experimental data are 300 pieces of treated Web texts and the updated experimental data are 150 pieces of treated Web texts, both of which have 5 keywords. After numerical labeled, all of the initial data are five-dimensional characteristic vectors.

The contrarest results between the CGDCP clustering algorithm and the K-means algorithm are listed in Table 1. Analyzing the above results, the performance of the CGDCP clustering algorithm is higher than the K-means algorithm.

<table>
<thead>
<tr>
<th>Text number</th>
<th>100</th>
<th>200</th>
<th>300</th>
</tr>
</thead>
<tbody>
<tr>
<td>CGDCP</td>
<td>Cr</td>
<td>94%</td>
<td>92%</td>
</tr>
<tr>
<td></td>
<td>Vr</td>
<td>89%</td>
<td>88%</td>
</tr>
<tr>
<td>K-means</td>
<td>Cr</td>
<td>90%</td>
<td>85%</td>
</tr>
<tr>
<td></td>
<td>Vr</td>
<td>88%</td>
<td>83%</td>
</tr>
</tbody>
</table>

The contrarest results between the ICECPG clustering algorithm and the K-means algorithm are listed in Table 2.

<table>
<thead>
<tr>
<th>Text number (300)</th>
<th>+50</th>
<th>+50</th>
<th>+50</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICECPG</td>
<td>Cr</td>
<td>91%</td>
<td>91%</td>
</tr>
<tr>
<td></td>
<td>Vr</td>
<td>87%</td>
<td>88%</td>
</tr>
<tr>
<td>K-means</td>
<td>Cr</td>
<td>81%</td>
<td>80%</td>
</tr>
<tr>
<td></td>
<td>Vr</td>
<td>76%</td>
<td>76%</td>
</tr>
</tbody>
</table>

Analyzing the above results, the performance of the ICECPG clustering algorithm is higher than the K-means algorithm. And the running time of the new incremental clustering algorithm is less than that of the K-means algorithm.

5. Conclusion
Recently, clustering has been recognized as a primary data mining method for knowledge discovery in spatial databases. In this paper, we introduce a new kind of clustering algorithm —— CGDCP. In order to make CGDCP applicable in periodically incremental environment, we also present an incremental clustering algorithm —— ICECPG, dealing with a bulk of updates instead of single update. Future work includes automatic determination of thresholds and improvement on the efficiency. Experimental results show that it can realize incremental clustering process very effectively and accurately. It can also deal with exceptional, high-dimension and complicated data.

References


