EXPECTED-TIME COMPLEXITY RESULTS FOR HIERARCHIC CLUSTERING ALGORITHMS WHICH USE CLUSTER CENTRES

F. MURTAGH

Department of Computer Science, University College Dublin, Dublin 4, Ireland

Communicated by L. Boasson
Received 24 November 1982
Revised 14 February 1983

Keywords: Probabilistic algorithms, dynamic searching, nearest neighbours

1. Introduction

Hierarchic clustering algorithms are finding increasing applications for structuring and interpreting data (see reviews in [1,2]). Their practical use depends, however, on fast algorithms for implementing them.

Much recent work has concentrated on the minimal spanning tree and the single link hierarchic clustering method which is closely related to it (see [3–5]).

Although fast algorithms exist for the single link method ([5] gives an O(N) expected time algorithm for the minimal spanning tree, and hence for the single link clustering), it suffers from two disadvantages. Firstly, the single link clustering has the ‘chaining’ effect of forming very unbalanced or skew hierarchies, and this restricts the usefulness of this method for summarizing data. Secondly, in the single link method there is no natural definition of ‘cluster representative’ or cluster centre.

Among widely used clustering methods which do not suffer from this second disadvantage are the median method (also known as Gower’s method or WPGMC), the centroid method (weighted group or UPGMC) and Ward’s minimum variance (or error sum of squares) method. All three define a cluster centre which replaces points which are agglomerated. Ward’s method is generally found to give a more balanced hierarchy than any other method.

Certain recent algorithmic improvements in these methods [6,7] have succeeded in making them computationally more efficient, and hence of greater practical interest. Performance results reported on in the last mentioned references are at best O(N^2) for the clustering of N points, since all pairwise distances must be examined at least once. We extend these results here by examining fast expected-time results. We show that for the median and centroid methods, where the cardinality of clusters is not used in the definition of dissimilarity between cluster centres, the expected-time performance is O(N). We also show that the expected time complexity of Ward’s method is upperbounded by O(N log N).

2. Hierarchic clustering algorithms

We consider N points in the Euclidean plane (results below hold best for this case, but can be generalized to other dimensions and dissimilarities). The traditional algorithm for hierarchic clustering may be formulated as follows.

Algorithm 1
Step 1. Determine all interpoint dissimilarities.
Step 2. Form cluster from two closest points.
Step 3. Redefine dissimilarities between new cluster and other points.

Step 4. Return to Step 2 until all points clustered into one cluster.

The dissimilarities in Step 1 will most often be distances (especially some Euclidean distance). In Step 3 the dissimilarity between the new point — replacing the two merged points — and any other point is given by the Lance–Williams recurrence formula (see [1]). An alternative algorithm, more suitable for clustering methods which replace points with a cluster centre, and also for speed-up algorithms which preprocess the points to be clustered into local, restricted regions, is as follows.

Algorithm 2

Step 1. Examine all interpoint dissimilarities, and form cluster from the two closest points.

Step 2. Replace the two points clustered with a cluster point.

Step 3. Return to Step 1 until all points clustered into one cluster.

An improvement in this algorithm, which allows the formation of as many clusters as possible in Step 1, is based on what may be termed the nearest neighbour (NN) graph. This is the set of vertices given by the points under consideration at any stage of the clustering, and the set of directed edges (i, NN(i)) where i is any point and NN(i) is the NN of point i. If j = NN(i) and i = NN(j), then i and j are called reciprocal NNs (RNNs). The set of all edges (i, NN(i)) in the NN-graph can be examined for RNNs in O(N) time, and if a pair of RNNs can be agglomerated independently of any other interpoint relationship, we have the following algorithm.

Algorithm 3

Step 1. Determine the NNs of all points.

Step 2. Find all RNNs, replacing each such pair of points with the cluster point.

Step 3. Return to Step 1 until all points are in the one cluster.

For further details regarding this algorithm, the reader is referred to [6,7].

At any stage in the clustering using Ward’s method, the agglomeration of a pair of RNNs causes no alteration of the NN relation for points which are not connected to them in the NN graph. Thus the hierarchies produced by Algorithms 1 and 3 for Ward’s method will be identical. On the other hand, the median and centroid hierarchies constructed by Algorithm 3 might differ marginally from the hierarchy constructed by Algorithm 1, due to the fact that the RNNs are considered in any order. It is possible that a new cluster centre could alter the NN-graph, and this is not allowed for in Algorithm 3. This aspect of the centroid and median methods — related to the possibility of non-monotonic variation in the cluster criterion values (inversions) — has little practical effect. In what follows we will ignore this small discrepancy.

An obvious variant of Algorithm 3 would only update the NN-graph in Step 1 to the extent that this was necessary; i.e., those points which were not agglomerated and whose NNs were not agglomerated, would not require a recalculation of their NNs. This variation was not found to improve computation time, and it renders analysis (see Section 3) more difficult. It may be noted that, in a Euclidean plane, up to 6 points might have a point as NN which gives an upper limit on the number of re-determinings of NNs for every agglomeration.

The definitions of cluster point, given a pair of RNNs in Step 2 of Algorithm 3, are as follows for three well-known hierarchical clustering methods (see [9, pp. 153–160]):

\[
\text{Median: } \frac{1}{2}(i+j),
\]

\[
\text{Centroid: } \frac{n_i + n_j}{n_i + n_j},
\]

\[
\text{Ward’s: } \frac{n_i + n_j}{n_i + n_j},
\]

where \(n_i\) and \(n_j\) are the numbers of members of the clusters represented by points i and j respectively. The definitions of dissimilarity, used in the calculation of NNs, for these methods are as follows:

\[
\text{Median: } \delta^2(i, j) = d^2(i, j),
\]
Centroid: \( d^2(i, j) = d^2(i, j), \)
\[ W^2(i, j) = \frac{n_i n_j}{n_i + n_j} d^2(i, j), \]
where \( d \) is the distance used.

3. Algorithm 3 with preprocessing

In this section we describe a preprocessing stage to Algorithm 3 which allows the finding of NNs to be carried out in faster than \( O(N) \) calculations per point. An assignment of the \( N \) points to a regular square grid is accomplished in \( O(N) \) time and if the input data is uniformly distributed, then the NNs of \( \) points are found in \( O(N) \) time.

**Hypothesis 1.** A set of points is uniformly and independently distributed in a bounded region of the plane.

**Proposition 1.** The NN of a point is found in constant expected time.

The NN search is carried out by a search of the grid square containing the point, and then in successive layers of grid squares adjacent to this, until the NN is found. A proof of this result is to be found in [5] or [10].

Two difficulties arise when applying this result to hierarchic clustering. Firstly, points representing clusters replace the initial points during the clustering process. There is no guarantee that they will be distributed in the same manner as the initial points, but making this hypothesis is motivated by the uniform selection of RNNs (in a similar manner to the uniform processing of minimal spanning tree fragments in Sollin’s algorithm).

**Hypothesis 2.** At each stage of the execution of Step 1 of Algorithm 3, the points (representing singletons or clusters) are uniformly distributed.

Secondly, having found the current NN of a point in some given layer of grid squares surrounding a point, a test must be applied to see if a potential NN could exist in some grid square in

the next, exterior layer. In the case of Ward’s method, this test will have to assume that the potential NN is a singleton point, i.e., a cluster of membership 1 (cf. the list of dissimilarity definitions at the end of Section 2). This lower bound estimate of the cluster membership of a potential NN can be a cause of inefficiency.

Because of this latter difficulty, the analysis of Algorithm 3 is most easily carried out in the case of the median or centroid methods, where a distance (rather than a dissimilarity using cluster membership) is used at all times. The following proposition is central in deriving the complexity of the median and centroid methods.

**Proposition 2.** Given \( N \) uniformly distributed points in the plane, a given point has an RNN with constant probability (approximately 0.6215).

This is proved in [11] or [8]. It follows from Proposition 2 that each re-entering of Step 1 in Algorithm 3 will be with \( x = (1 - \frac{1}{2} \times 0.6215) \) times the number of points on the previous execution of this step.

The computational requirements of Steps 1 and 2 of Algorithm 3 are directly proportional to the number of points under consideration. (This includes the hashing of the points under consideration at each execution of Step 1 in order to preserve a constant density of points per grid square.) Therefore, the overall number of computations required in Algorithm 3 is directly proportional to \( N + xN + x^2N + x^3N + \cdots \), which is \( O(N) \). We thus have the following proposition.

**Proposition 3.** Under Hypotheses 1 and 2, the expected time complexity of Algorithm 3 for the centroid and median methods is \( O(N) \).

Because Ward’s minimum variance method, unlike the centroid and median methods, uses a dissimilarity between points representing clusters, which takes into account the cardinality of these clusters, we will obtain an upper bound on the expected time by considering a ‘worst-case hierarchy’. In Ward’s method, on finding a current
NN point \( i' \) to a given point \( i \), the following test must be true in order that no potential NN exists in some layer of grid squares exterior to the current layer:

\[
\frac{n_i n_{i'}}{n_i + n_{i'}} \cdot d^2(i, i') \leq \frac{n_i}{n_i + 1} d^2(i, \cdot), \tag{*}
\]

where \( d^2(i, \cdot) \) is the minimum distance from \( i \) to the exterior of the region of grid squares currently being searched. A sufficient condition for the above inequality to hold is that

\[n_i, d^2(i, i') \leq d^2(i, \cdot),\]

i.e., that \( n_i \) times the NN-distance squared be examined. This means that if \( i' \) is in fact the NN of \( i \), then we must nevertheless verify that no point could be NN in a region which is \( n_i \) times the region just examined; i.e., that the number of points examined is \( n_i \) times the number required to find \( i' \) (and the latter – the expected number of points to be examined in order to find \( i' \) – is constant by virtue of Proposition 1).

In (\( * \)), \( d^2(i, \cdot) \) is greatest if, given \( n_i + n_{i'} \), \( n_i \) and \( n_{i'} \) are approximately equal. We thus have the following proposition.

**Proposition 4.** The hierarchic clustering for which the greatest number of redundant calculations of dissimilarity are carried out, is of the following form:

- \( N \) points agglomerate to form \( \frac{1}{2}N \) points,
- \( \frac{1}{2}N \) points agglomerate to form \( \frac{1}{4}N \) points,
- \( \frac{1}{4}N \) points agglomerate to form \( \frac{1}{8}N \) points,
- etc.

At the stage when we have \( \frac{1}{2}N \) points, \( n_i \), as used above equals \( 2 \) and twice the number of points will have to be examined to find the NN of each of these \( \frac{1}{2}N \) points than would be the case in using a Euclidean distance. At the stage when we have \( \frac{1}{4}N \) points, \( 4 \) times the number of points will be examined. Overall, the number of calculations is directly proportional to

\[N + 2 \times \frac{1}{2}N + 4 \times \frac{1}{4}N + 8 \times \frac{1}{8}N + \cdots .\]

There are \( \log N \) such terms. We thus upperbound the time performance of Ward’s method for the construction of any hierarchy.

<table>
<thead>
<tr>
<th>N</th>
<th>CPU time</th>
<th>Const. N log N</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td>17.73</td>
<td>17.73</td>
</tr>
<tr>
<td>4000</td>
<td>37.37</td>
<td>38.65</td>
</tr>
<tr>
<td>6000</td>
<td>57.33</td>
<td>60.81</td>
</tr>
<tr>
<td>8000</td>
<td>78.67</td>
<td>83.86</td>
</tr>
<tr>
<td>10000</td>
<td>99.68</td>
<td>107.44</td>
</tr>
</tbody>
</table>

**Proposition 5.** Under Hypotheses 1 and 2, the expected time complexity of Algorithm 3 for Ward’s method is upperbounded by \( O(N \log N) \).

4. Results and conclusion

Empirical results obtained using the median method, and uniformly distributed points in the Euclidean plane, are given in Table 1, where CPU times are in seconds, averaged over 5 replications, on a DEC-20 machine. These times increase marginally faster than \( O(N) \), due to the approximate nature of Hypothesis 2.

The empirical results obtained for Ward’s method are given in Table 2, where it is clear that the order of magnitude is bounded by \( O(N \log N) \). Again, averages of 5 replications are given, and the rightmost column is \( N \log N \) multiplied by a suitable constant so that for \( N = 2000 \) values in columns 2 and 3 are equal.

In conclusion, although the case of 2-dimensional data only has been discussed, results obtained also hold for higher dimensions but since the numbers of grid cells adjacent to a given cell grow exponentially with dimensionality, the pract...
tical use of the discussed preprocessing approach is limited to small dimensions. The Euclidean distance, perhaps weighted, is most likely to be used with the particular hierarchical clustering procedures which have been discussed. Finally, although uniformly distributed data is unlikely to be obtained in practical applications, extensions of the algorithms discussed could allow for irregular grids of varying sized squares (cells) in order to verify Proposition 1 for different data configurations (see [5]).

References