SIHC: A STABLE INCREMENTAL HIERARCHICAL CLUSTERING ALGORITHM

Ibai Gurrutxaga, Olatz Arbelaitz, José I. Martín, Javier Muguerza, Jesús M. Pérez and Iñigo Perona
Computer Architecture and Technology Dept., University of the Basque Country, Donostia, Spain

Keywords: Hierarchical clustering, Incremental, Stability.

Abstract: SAHN is a widely used agglomerative hierarchical clustering method. Nevertheless it is not an incremental algorithm and therefore it is not suitable for many real application areas where all data is not available at the beginning of the process. Some authors proposed incremental variants of SAHN. Their goal was to obtain the same results in incremental environments. This approach is not practical since frequently must rebuild the hierarchy, or a big part of it, and often leads to completely different structures. We propose a novel algorithm, called SIHC, that updates SAHN hierarchies with minor changes in the previous structures. This property makes it suitable for real environments. Results on 11 synthetic and 6 real datasets show that SIHC builds high quality clustering hierarchies. This quality level is similar and sometimes better than SAHN's. Moreover, the computational complexity of SIHC is lower than SAHN's.

1 INTRODUCTION

Clustering is an unsupervised pattern classification method which partitions the input space into groups or clusters. The goal of a clustering algorithm is to perform a partition where objects within a group are similar and objects in different groups are dissimilar. Therefore, the purpose of clustering is to identify natural structures in a dataset and it is widely used in many fields (Jain and Dubes, 1988; Mirkin, 2005; Sneath and Sokal, 1973).

Many clustering algorithms exist, but all of them can be broadly classified in two groups: partitional and hierarchical. Partitional algorithms process the input data and create a partition that groups the data in clusters. On the other hand, hierarchical algorithms build a nested partition set called cluster hierarchy. The procedures to obtain this nested set of partitions are classified as agglomerative (bottom-up) and divisive (top-down).

These clustering hierarchies are suitable to present in a graphical way. The most usual procedure is to draw them as a tree diagram called dendrogram. The root node of this tree contains the topmost partition while the leaf nodes contain the partition at the lower level of the hierarchy. A merge in an agglomerative algorithm is represented as a new node which is set to be the parent of the merged nodes. Similarly, a split in a divisive algorithm is viewed as new nodes which are set to be the children of the split node. This graphical representation is very helpful for the end user since it allows a rapid identification of the most interesting structures in the data.

Several hierarchical algorithms exist (Fisher, 1987; Jain and Dubes, 1988; Mirkin, 2005). SAHN is a widely known general method to build cluster hierarchies (Sneath and Sokal, 1973). It defines a simple and intuitive procedure and has been widely used in many areas. Unfortunately it is not an incremental method, meaning that we must rebuild the entire hierarchy if we want to add some new data to an existing one. Since many real applications work with a data stream some authors proposed incremental versions of the SAHN method (El-Sonbaty and Ismail, 1998; Ribert et al., 1999). The goal of these approaches is to incrementally build a tree identical to the one that would be obtained with the original SAHN method. This goal is interesting from a theoretic point of view, but we claim it is not crucial from a practical point of view.

Final users of incremental clustering algorithms need a stable structure. The user updates its knowledge as the incremental procedure is carried out. Consequently this process must keep the main structure of the updated dendrogram. If the addition of few new cases leads to a dramatic change of the learned

Gurrutxaga I., Arbelaitz O., Martín J., Muguerza J., Pérez J. and Perona I. (2009).
SIHC: A STABLE INCREMENTAL HIERARCHICAL CLUSTERING ALGORITHM.
In *Proceedings of the 11th International Conference on Enterprise Information Systems - Artificial Intelligence and Decision Support Systems*, pages 300-304
DOI: 10.5220/0001857103000304

structures, the user will be unable to assimilate the changes. In addition, it will lose confidence on the learning algorithm. The incremental versions of the SAHN algorithm do not provide a stable structure. Since their goal is to obtain the same tree we would obtain with the original method they must, in many cases, rebuild the entire tree or an entire branch of it.

The purpose of our work is to present a stable incremental version of SAHN algorithm. The proposed algorithm adds a new case without varying the previously obtained structures.

To evaluate the proposed algorithm we answer two main questions in this paper. Which is the quality of the incremental dendrograms compared to those built with the SAHN method? How does the arrival order of the new cases affect the updating algorithm? We experimented with 11 synthetic and 6 real datasets and 3 algorithms based on the SAHN method. Results confirm that the incremental algorithm is able to update dendrograms with no quality loss. Therefore, we claim this algorithm is a good choice on real incremental environments where stability and comprehensibility are important factors.

Next section is devoted to describe the SAHN method and some previous incremental approaches. The algorithm we propose is described in Section 3. In Section 4 we describe the experimentation designed to measure the quality of the proposed algorithm and results are described in Section 5. Finally, Sections 6 and 7 are devoted to discuss the results and draw conclusions.

2 HIERARCHICAL CLUSTERING

In this section we first describe the SAHN method: a method to obtain cluster hierarchies. Then, we briefly describe two previously published incremental versions.

The SAHN method is based on two main steps. First, each individual point of the input dataset forms a cluster on its own. That is, if n denotes the number of data points in the dataset the method begins with a partition of n singleton clusters. Second, the two closest clusters are merged. This second step is repeated until all the points are merged in a single cluster. This method automatically obtains a set of nested partitions forming a hierarchy. The output of this procedure is a set of exactly n partitions, from the n singleton clusters to the all-in-one cluster partition.

To measure the proximity of two clusters we must define a cluster proximity measure. The SAHN method is often used with one of the following proximity measures: single-linkage (nearest neighbour), complete-linkage (farthest neighbour) and average-linkage (group average). Single-linkage computes the distance between two clusters as the distance between the two nearest points in both clusters. Similarly, complete-linkage computes the distance between the two farthest points in both clusters. Finally, average-linkage computes the average distance between all the points from one cluster to the points in the other cluster.

(Ribert et al., 1999) proposed an incremental variation of the SAHN method. The tree they obtain is exactly the same obtained applying the SAHN method to the union of the initial dataset and the new case. Although an analysis of the computational cost of the incremental method is not done, empirical results show similar behaviour to SAHN's. On the other hand, the memory usage is considerably reduced.

(El-Sonbaty and Ismail, 1998) described another incremental version of the single-linkage algorithm. Its computational cost is $O(n^2)$, opposite to the $O(n^3)$ cost of the SAHN method. They empirically showed that the algorithm obtains the same tree as the single-linkage algorithm for some values of an internal parameter of the algorithm. Nevertheless, the new method cannot be applied to other proximity measures commonly used with the SAHN method.

3 SIHC ALGORITHM

The aim of the algorithm we present, called SIHC, is to incrementally add new cases to dendrograms built with the SAHN method. The novelty of this incremental algorithm is that the updated tree keeps its main structure. This way users will easily assimilate the small variations produced by the learned cases. Although the algorithm can be used on its own, we designed it as an updating method for SAHN-based dendrograms.

SIHC is a top-down process described in Algorithm 1. It is a recursive procedure that begins at the root node. This procedure computes the distance between the new case and the cluster represented by the current node. This distance can be based on any proximity measure, so the method adapts to any SAHN-based algorithm. If the height of the node is less or equal than the computed distance the recursive procedure stops. A new node, whose children are the new case and the current node, is created and it replaces the current node in the dendrogram.

If the height of the current node is higher than the distance from the new case to it, the recursive procedure is repeated on the child nearest to the new case. In this case the height of the traversed node should be

Algorithm 1 : SIHC.

$d_{node} \leftarrow$ distance$(newcase, node)$
if $node.height \leq d_{node}$ **then**
  $singleton \leftarrow$ new leaf node
  $singleton$.addCase$(newcase)$
  $newnode \leftarrow$ new internal node
  $newnode$.addChildren$(node, singleton)$
  $newnode.height \leftarrow d_{node}$
  replace $node$ with $newnode$
else
  $nearest \leftarrow \arg\min_{child}$(distance$(newcase, child)$)
  update_height$(node)$
  SIHC$(newcase, nearest)$
end if

updated. The new height is easily computed since it is the distance between its children once the new case is added to the nearest children. Anyway, for most used proximity measures there is a simple way to compute this value. Let h be the height of the traversed node, n_{near} the number of cases in the child nearest to the new case and d_{far} the distance from the new case to the farthest child. The new height of the node is computed as follows:

- Single-linkage: $\min(h, d_{far})$
- Complete-linkage: $\max(h, d_{far})$
- Average-linkage: $(h \times n_{near} + d_{far})/(1 + n_{near})$

Similar functions can easily be found for other proximity measures.

The algorithm ensures that the addition of a new case will not vary previously defined structures. It just adds a new node where appropriate. Notice that the definition of the procedure prevents incoherences such as a node being located higher than its parent node.

4 EXPERIMENTAL SETUP

In this section we describe the experimental work performed to measure the quality of the dendrograms updated with the SIHC algorithm. We ran the algorithm over 11 synthetic and 6 real datasets as explained next. We split each dataset in 20 folds and built a dendrogram using the SAHN method and k folds. We incrementally added the remaining $20 - k$ folds based on SIHC algorithm. k varied from 1 to 19, and we used three proximity measures: single, average and complete linkage. This procedure allowed us to measure how variation on the fraction of incrementally learned cases affects the obtained dendrogram.

Table 1: Characteristics of the real datasets.

Dataset	Dimensions	Clusters	Cases
Iris	4	3	150
Glass	9	7	214
Wine	13	3	178
Ecoli	8	8	336
Haberman	3	2	306
Ionosphere	34	2	351

To measure the effect produced by the arrival order of the data we repeated the procedure 25 times. We built 5 dendrograms for each k value randomly selecting the k folds. Furthermore, we incrementally added the remaining cases in 5 different orders. This means that we built 25 incremental dendrograms for each dataset, k value and proximity measure. In order to have a valid reference we also built a SAHN tree with all the data from each dataset, resulting in 24,276 dendrograms altogether.

We draw the 6 real datasets from the UCI repository (Asuncion and Newman, 2007). Their characteristics are described in Table 1. 9 of the synthetic datasets we designed follow the same pattern and are named as *Normal_d_s*. Here d means the number of dimensions and we set it to 2, 4 and 6. In each dataset there is a cluster per dimension created drawing 50 points each from a multivariate normal distribution. The mean of the distribution of cluster i is set to $s * e_i$ and the covariance matrix of every cluster is the identity matrix. e_i is the vector where i^{th} position is set to 1 while the rest is set to 0. We set s to 3, 5 and 10, and hence obtained clusters with different overlapping level. The remaining two datasets, *Concentric* and *T&U*, are 2-dimensional and define clusters that are hard to detect for many clustering algorithms. The former contains 3 concentric ring-shaped clusters and a total of 400 points. The latter is composed by one T-shaped and one U-shaped cluster, of 150 points each. The trunk of the T is in the concave part of the U.

To measure the quality of a dendrogram we prefered the partition membership divergence (PMD) distance rather than the widely used cophenetic distance because "identical topologies may still prove very different in cophenetic levels" (Podani, 2000). PMD is defined as the number of partitions implied by the dendrogram in which a given pair of cases do not belong to the same cluster. We measured the quality of a dendrogram computing the correlation between the PMD matrix of the dendrogram and the distance matrix.

Although the aim of the proposed algorithm is not to obtain the same dendrogram SAHN would

build, we consider interesting to compare the dendrograms obtained by both algorithms. We compared two dendrograms computing the correlation between their PMD matrices.

5 RESULTS

In Figure 1 we show the quality of the dendrograms obtained by SIHC. In order to have a valid reference all values were divided by the quality of the corresponding SAHN-based dendrogram. This means that a value greater than 1 represents higher quality than the SAHN-based dendrogram and a value lower than 1 represent lower quality. Remind that for each dataset, algorithm and k value we computed 25 dendrograms. Each result showed in Figure 1 refers to the average value of those 25 runs.

The left side of the figure shows average results for the synthetic datasets. It is clear that the quality of the dendrogram increases with k value. This means that better dendrograms were obtained when less cases were incrementally added. Incremental dendrograms based on single-linkage never improve the quality of the SAHN-based dendrogram. On the other hand, those based on complete and average-linkage achieve improvements, provided that no more than 30% or 20% of the data was added incrementally. If we focus on real datasets — on the right side of the figure— results for single-linkage are better, improving SAHN-based dendrograms for 5 k values. For complete and average-linkage results are similar to those obtained for synthetic datasets, but they are slightly better for low k values and slightly worse for high k values. Anyway, the quality of dendrograms with about 25% of incrementally added data is higher than the quality of the corresponding SAHN dendrograms. All individual datasets show a similar behaviour pattern.

Previous results suggest that incremental dendrograms differ from SAHN dendrograms, at least those built with average or complete linkage. We have measured this dissimilarity relation comparing all incremental dendrograms to their corresponding SAHN dendrogram. Results show that similarity increases with k. SIHC is very similar to SAHN for single-linkage while for average and complete-linkage high similarity levels are obtained for just the highest k values. Nevertheless, we saw that this dissimilarities often allowed a quality increment.

Since the result for each dataset, algorithm and k value was the average value of 25 executions we computed the standard deviation of the quality of the dendrograms— the correlation of the PMD matrix of each dendrogram and the distance matrix. The goal of this analysis was to measure the stability of SIHC faced to modifications in the arrival order of the data. Results show that the variation is insignificant. Average standard deviation was below 0.04 and just a few cases exceeded 0.15. No significant pattern could be found in these results although it seems that variance decreases when k increments.

6 DISCUSSION

The results of the experimentation show that SIHC algorithm builds high-quality dendrograms. This quality is even higher than SAHN's provided that no more than a specific fraction of the total number of cases have been incrementally added. This threshold is database and proximity measure dependant, but is about 35%. Results confirm that SIHC can be used on its own, but is better used as an updating method for SAHN dendrograms. Nevertheless, if single-linkage proximity measure is used, results for SIHC are similar to SAHN's even in stand-alone mode. The algorithm works in optimal conditions if it is used to update a SAHN dendrogram with a low flow of new data.

Results also show that the incrementally updated dendrograms differ from the SAHN dendrograms, particularly if average or complete-linkage is used. The positive point is that the differences produced by SIHC are often used to improve SAHN's results.

Regarding to the computational cost of SIHC a worst case analysis determines a $O(n^2 \times \log(n))$ complexity. Notice that each case must traverse, at worst case, a whole branch of the tree from the root to a leaf node. The length of this path is $O(\log(n))$. In each node the new case must be compared to the cases already in it, which is $O(n)$. Since n is the total number of cases the overall complexity is $O(n^2 \times \log(n))$. This complexity level is sensitively lower than SAHN's $O(n^3)$ complexity.

7 CONCLUSIONS

In this work we presented SIHC: a new incremental algorithm based on SAHN method. This algorithm allows to incrementally add new cases to a previously built dendrogram. The novelty of the algorithm is that it adds the new data with no changes in the main structure of the updated dendrogram. It simply adds a new node were appropriate. This stability is fundamental for many practical purposes and was ignored in previous incremental approaches. Furthermore, the loca-

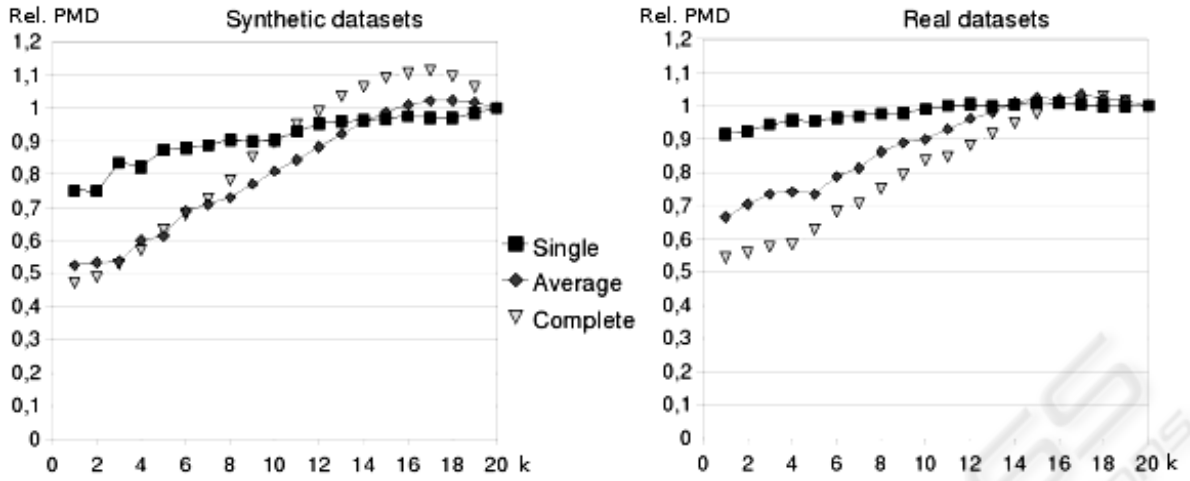

Figure 1: Quality of the incremental dendrograms.

tion of the added node can be used with classification purposes, since it determines the membership of the new case to previously detected structures or clusters.

Results show that the proposed algorithm builds similar dendrograms to SAHN for single-linkage, but differences arise with other proximity measures. Nevertheless, these differences allow SIHC to build better quality dendrograms. It seems that these improvements are kept while the incrementally added data is about half of the data in the initial SAHN tree.

The incremental algorithm can be adapted to any proximity measure used with SAHN method. Moreover, many computations can be avoided based on properties that the most used proximity measures have. On the other hand, although SIHC depends on how the incremental data is ordered, variations are not significant.

We also showed that the complexity of SIHC significantly reduces the complexity of SAHN method from $O(n^3)$ to $O(n^2 \times \log(n))$.

In few words, we present a new algorithm that allows updating SAHN dendrograms, at a reduced computational cost, keeping the main cluster structures. This capabilities make the algorithm suitable for practical use in real enterprise environments.

ACKNOWLEDGEMENTS

The work described in this paper was partly done under the University of the Basque Country, project EHU 08/39. It was also funded by the Diputacin Foral de Gipuzkoa and the European Union.

REFERENCES

Asuncion, A. and Newman, D. (2007). UCI machine learning repository. http://www.ics.uci.edu/∼mlearn/MLRepository.html

El-Sonbaty, Y. and Ismail, M. (1998). On-line hierarchichal clustering. *Pattern Recognition Letters*, 19:1285–1291.

Fisher, D. H. (1987). Knowledge acquisition via incremental conceptual clustering. *Machine learning*, 2:139–172.

Jain, A. K. and Dubes, R. C. (1988). *Algorithms for Clustering Data*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Mirkin, B. (2005). *Clustering for Data Mining: A Data Recovery Approach*. Chapman & Hall/CRC.

Podani, J. (2000). Simulation of random dendrograms and comparison tests: Some comments. *Journal of Classification*, 17:123–142.

Ribert, A., Ennaji, A., and Lecourtier, Y. (1999). An incremental hierarchical clustering. In *Vision Interface '99*, pages 586–591, Trois-Rivières, Canada.

Sneath, P. H. A. and Sokal, R. R. (1973). *Numerical Taxonomy*. Books in biology. W. H. Freeman and Company.