A General Framework for Agglomerative Hierarchical Clustering Algorithms

Reynaldo J. Gil-García[1], José M. Badía-Contelles[2] and Aurora Pons-Porrata[1]
[1]Center of Pattern Recognition and Data Mining, Universidad de Oriente, Cuba
[2]Universitat Jaume I, Spain
{gil,aurora}@csd.uo.edu.cu, badia@icc.uji.es

Abstract

This paper presents a general framework for agglomerative hierarchical clustering based on graphs. Specifying an inter-cluster similarity measure, a subgraph of the β-similarity graph, and a cover routine, different hierarchical agglomerative clustering algorithms can be obtained. We also describe two methods obtained from this framework called Hierarchical Compact Algorithm and Hierarchical Star Algorithm. These algorithms have been evaluated using standard document collections. The experimental results show that our methods are faster and obtain smaller hierarchies than traditional hierarchical algorithms while achieving a comparable clustering quality.

1. Introduction

Hierarchical clustering solutions have an additional interest for a number of application domains, because they provide a view of the data at different levels of abstraction, making them ideal for people to visualize and interactively explore large collections. Besides, clusters very often include subclusters, and the hierarchical structure is indeed a natural constraint on the underlying application domain (e.g. biological or documental taxonomies). In some cases, like in the information organization problems, it is desirable to obtain overlapped hierarchies, since documents can deal with multiple topics.

Many hierarchical clustering algorithms have been proposed, for example, *Complete-link*, *Average-link* and *Bisecting K-Means* [5]. Since a pair of clusters are only merged at each iteration (or a cluster is split into two subclusters), these algorithms produce large hierarchies. Thus, they spend a lot of time recalculating the similarities between the new cluster and all remaining clusters in each level of the hierarchy.

Recently, Yu et al. [7] proposed an algorithm for hierarchical topic detection [6]. It is based on a multi-layered clustering to produce the hierarchy, by applying successively the *Single-Pass* algorithm. It starts at a certain threshold to cluster in the bottom layer and then, the threshold is decreased in the next levels until the root is generated. The algorithm requires to tune several parameters and the obtained clusters depend on the data order.

Following the idea of multi-layered clustering, we propose a general framework of hierarchical agglomerative clustering algorithms. Specifying an inter-cluster similarity measure, a subgraph of the β-similarity graph, and a cover routine, different hierarchical agglomerative clustering algorithms can be obtained. All such methods share three main features. First, they are based on graphs. This property guarantees not only that any similarity measure can be used (not necessarily a metric), but also allows the algorithms to handle mixed objects, that is, described by numerical and categorical attributes. Second, they could obtain disjoint or overlapped hierarchies depending on the cover routine used. Finally, the obtained hierarchies have few levels because several clusters can be merged in each level.

Two specific algorithms obtained from this framework: *Hierarchical Compact Algorithm* and *Hierarchical Star Algorithm* are also introduced. The first creates disjoint hierarchies of clusters, while the second obtains overlapped hierarchies. Both algorithms require a unique parameter and the obtained clusters are independent on the data order.

2. Hierarchical Agglomerative Framework

We call *β-similarity graph* the undirected graph whose vertices are the clusters and there is an edge from vertex i to vertex j, if the cluster j is β-similar to i. Two clusters are *β-similar* if their similarity is greater or equal to β, where β is a user-defined parameter. Analogously, i is a *β-isolated cluster* if its similarity with all clusters is less than β.

The clustering algorithms based on graphs involve two main tasks: the construction of a certain graph and a cover routine of this graph that determines the clusters. In this context, a cover for a graph $G = (V, E)$ is a collection $V_1, V_2, ..., V_k$ of subsets of V such that $\cup_{i=1}^{k} V_i = V$, each one representing a cluster.

Our framework is an agglomerative method and it is also based on graphs. It uses a multi-layered clustering to produce the hierarchy. The granularity increases with the layer of the hierarchy, with the top layer being the most general and the leaf nodes being the most specific. At each successive layer of the hierarchy, vertices represent subsets of their parent clusters. The process in each layer involves three steps: the construction of the β-similarity graph, the construction of its subgraph, and the obtaining of the cover of the subgraph. The general framework is shown in Fig. 1.

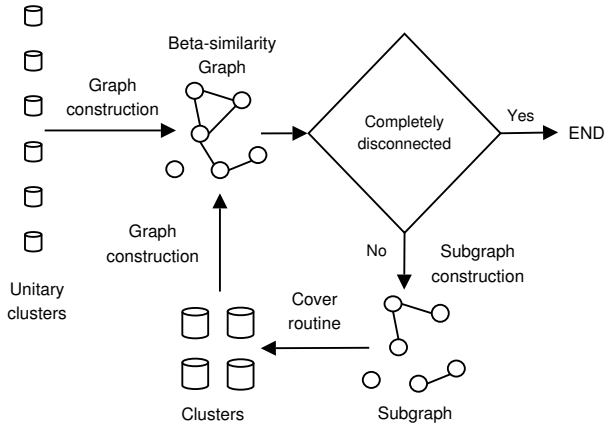

Figure 1. General framework.

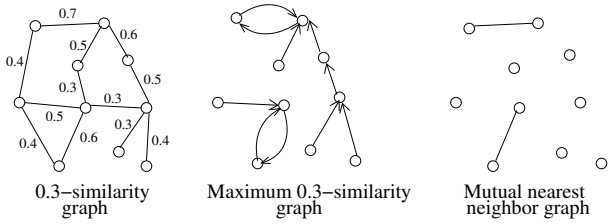

Figure 2. Graphs based on β-similarity.

In our framework, a similarity measure to compare the objects and an inter-cluster similarity measure are required. The algorithm starts with each object being considered a cluster. Then, it constructs the β-similarity graph and its subgraph. The set of vertices of this subgraph must be equal to the set of vertices of the graph. A cover routine is applied to this subgraph in order to build the clusters in the bottom layer. From the obtained clusters, the algorithm builds a new β-similarity graph and its corresponding subgraph. In these graphs, the vertices represent the clusters of the previous layer, and the edges are obtained using the inter-cluster similarity measure. Then, the cover routine is applied again to obtain the clusters in the next layer. This process is repeated until the β-similarity graph is completely disconnected, that is, all vertices (clusters) of the graph are β-isolated. Notice that we use the same β value and a unique subgraph type in all levels of the hierarchy.

We can obtain disjoint or overlapped clusters at each level of the hierarchy, depending on the cover routine used. It is worth noticing that if we change the type of subgraph, the similarity measures or the cover routine, different hierarchical agglomerative algorithms are obtained from this framework.

Single-link, *Complete-link* and *Average-link* methods can be seen as particular cases of the previous general framework. Those methods can be obtained from the framework if we choose $\beta = 0$, the subgraph of the β-similarity graph should be the mutual nearest neighbor graph (see Figure 2), and the cover routine should find the connected components in this subgraph. It is worth mentioning that in these cases, the obtained clusters will not depend on the data order, because in our algorithm all pairs of clusters with the maximum similarity are merged at the same time at each iteration.

3. Specific algorithms

In this paper, we propose two specific algorithms obtained from the abovementioned framework. Both algorithms use the maximum β-similarity graph. We call *maximum β-similarity graph* to the oriented graph whose vertices are the clusters and there is an edge from vertex i to vertex j, if the cluster j is the most β-similar to i. This graph is obtained from the β-similarity graph (see Figure 2).

The first method is *Hierarchical Compact Algorithm* (*HCA*). It assumes the following issues: 1) the subgraph is the maximum β-similarity graph disregarding the orientation of its edges (denoted as *U-max-S* graph), 2) the cover routine finds the connected components of the *U-max-S* graph, that is, the compact sets [4] (see Figure 3), and 3) it uses the group-average as inter-cluster similarity. Notice that, in this case, the clusters at each level of the hierarchy are connected components. For that reason, the obtained hierarchy is composed by disjoint clusters.

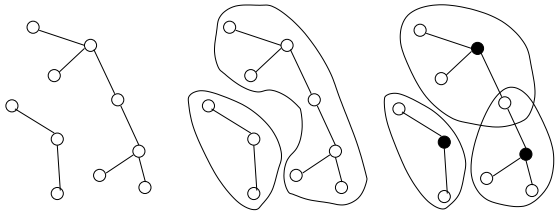

Figure 3. Cover routines of the *HCA* and *HSA* algorithms.

The main steps of the *HCA* algorithm are:
1. Put each object in a cluster on its own.
2. $level = 0$.
3. Construct the β-similarity graph, G_{level}.
4. While G_{level} is not completely disconnected:

(a) Construct the *U-max-S* graph (subgraph of G_{level}).

(b) Find the connected components of this subgraph.

(c) Construct a new β-similarity graph, $G_{level+1}$.

(d) $level = level + 1$

The second algorithm obtained from our framework is *Hierarchical Star Algorithm (HSA)*. It differs from the *HCA* algorithm in that it uses as cover routine at the step 4(b) the Extended Star cover [1] (see Figure 3). This cover routine approaches the minimum dominating set of the *U-max-S* graph using a greedy heuristic that takes into account the number of non-covered neighbors of each object. Each cluster is a star-shaped subgraph of *l*+1 vertices. It consists of a single star and l satellite vertices, where there exist edges between the star and each satellite vertex. The extended star cover of the *U-max-S* graph can be obtained as follows:

4.(b) While a non-covered vertex exists:

i. Let M_0 be the set of vertices with maximum number of non-covered neighbors.

ii. Let M be the subset of vertices of M_0 with minimum degree. M contains the stars.

iii. Create a cluster with each object of M and its neighbors and add them to the cover.

This cover does not present the chaining effect so common when connected components are used. However, it is worth mentioning that the *HCA* algorithm neither has a large chaining effect due to the use of the maximum β-similarity graph. Also, notice that the cover routine in the *HSA* algorithm creates overlapped clusters. The construction of overlapped hierarchies is a relevant feature of this algorithm.

The two proposed algorithms can produce clusters with arbitrary shapes, as opposed to algorithms such as *Bisecting K-Means*, which require central measurements in order to generate the clusters, restricting the shapes of these clusters to be spherical. Also, since we use the maximum β-similarity graph, our algorithms produce very cohesive clusters. The use of group-average as inter-cluster similarity measure avoids the problem of inversions in the cluster hierarchy. Another advantage is that they require a unique parameter and therefore, thus reducing the problem of tuning the parameter values to suit specific applications.

The time complexity of both algorithms is $O(n^2)$ and it is determined by the β-similarity graph construction. Each inter-cluster similarity is calculated using the Lance-Williams updating formula [2]. The space requirement for the algorithms is $O(n^2)$, but it strongly depends on the β value. In practice, $\beta > 0$ and we discard many pairwise similarities, greatly reducing the spatial cost of the algorithm.

4. Experimental results

The performance of the *HCA* and *HSA* algorithms has been evaluated using four standard document collections, whose general characteristics are summarized in Table 1. Human annotators identified the topics in each collection.

Table 1. Description of collections.

Collection	Source	Documents	Terms	Topics
AFP	TREC-5	695	12575	25
ELN	TREC-4	5829	84344	50
TDT	TDT2	9824	55112	193
REU	Reuters-21578	10369	35297	120

In our experiments, the documents are represented using the traditional vectorial model. The terms of documents represent the lemmas of the words appearing in the texts. Stop words, such as articles, prepositions and adverbs are disregarded from the document vectors. Terms are statistically weighted using the term frequency (TF). To account for documents of different lengths, the vector is normalized using the document length. We use the traditional cosine measure to compare the documents.

There are many different measures to evaluate the quality of clustering. We adopt a widely used external quality measure: the *Overall F-measure* [3]. This measure compares the system-generated clusters with the manually labelled topics and combines the precision and recall factors. The higher the overall F-measure, the better the clustering is, due to the higher accuracy of the clusters mapping to the topics. Our experiments were focused on evaluating the quality of the clustering produced by other well known hierarchical clustering methods: *Average-link*, *Complete-link* and *Bisecting K-Means*. We compare these methods with *HCA* and *HSA* algorithms.

The results for the document collections are shown in Table 2. In our algorithms we only evaluated the top level of the hierarchy and the parameter β that produced the best results was chosen. We can do that, because our methods have a well-defined stop condition. On the contrary, in the other algorithms we consider the flat partition produced by the best level of the hierarchy.

Despite that the best level selection can benefit traditional algorithms, our methods are either the best or always near to the best solution. It is worth mentioning that our algorithms obtain these results with both less total number of clusters and levels, thus the obtained hierarchy is easier to browse. In particular, *HSA* algorithm obtains slightly larger hierarchies than *HCA* algorithm, because the extended star clusters are subsets of the connected components in the *U-max-S* graph.

Table 2. Quality results.

Data	Algorithm	Levels	Clusters	Overall F
AFP	*Average-link*	695	1389	0.84
	Complete-link	695	1389	0.83
	Bisecting K-Means	695	1389	0.69
	HCA ($\beta = 0.12$)	3	226	0.82
	HSA ($\beta = 0.13$)	5	496	0.83
ELN	*Average-link*	5829	11658	0.41
	Complete-link	5829	11658	0.41
	Bisecting K-Means	5829	11658	0.36
	HCA ($\beta = 0.10$)	4	1033	0.46
	HSA ($\beta = 0.12$)	7	3638	0.46
TDT	*Average-link*	9824	19645	0.77
	Complete-link	9824	19645	0.50
	Bisecting K-Means	9824	19645	0.40
	HCA ($\beta = 0.12$)	4	2636	0.76
	HSA ($\beta = 0.13$)	7	6335	0.76
REU	*Average-link*	10369	20737	0.53
	Complete-link	10369	20737	0.37
	Bisecting K-Means	10369	20737	0.23
	HCA ($\beta = 0.12$)	4	2095	0.52
	HSA ($\beta = 0.11$)	8	6019	0.52

Figure 4 shows the time spent by our algorithms and three classical hierarchical algorithms mentioned above. Each curve represents the time spent to cluster the document sub-collections of sizes 1000, 2000 and so on. As we can observe, the *HCA* and *HSA* algorithms are faster than the other algorithms.

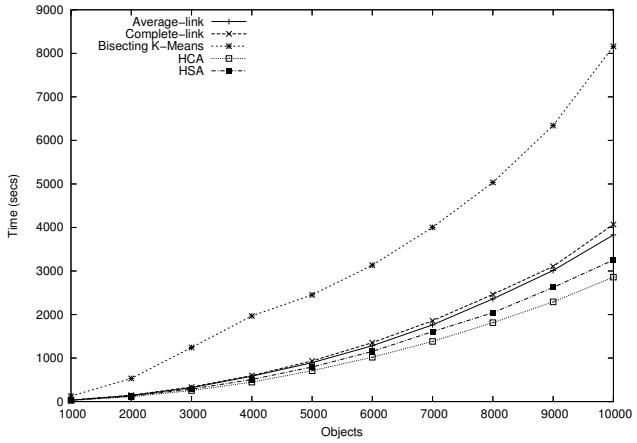

Figure 4. Time performance.

5. Conclusions

In this paper, a hierarchical agglomerative clustering framework based on the β-similarity graph has been presented. Different hierarchical agglomerative algorithms can be obtained from it, by specifying an inter-cluster similarity measure, a subgraph of the β-similarity graph, and a cover routine of this subgraph. The traditional hierarchical agglomerative methods can be seen as particular cases of this general framework. One of the most relevant features of the framework is that it allows obtaining disjoint or overlapped hierarchies composed by few levels.

Two specific variants of the proposed framework, called *Hierarchical Compact Algorithm* and *Hierarchical Star Algorithm* are also introduced. These algorithms obtain cohesive clusters with arbitrary shapes and they require a unique parameter. The *Hierarchical Star Algorithm* has also an important novelty: it obtains overlapped cluster hierarchies.

In the experiments with four document collections our algorithms obtain similar clustering quality than other traditional hierarchical algorithms. However, our methods obtain these results with less number of levels and clusters than other algorithms, thus the hierarchies are smaller and easier to browse. Besides, our methods are faster than other traditional algorithms.

Therefore, we advocate its use for tasks that require hierarchical clustering, such as creation of document taxonomies and hierarchical topic detection. Although we employ our algorithms to cluster document collections, they can be also applied to any problem of Pattern Recognition with mixed objects.

References

[1] R. J. Gil-García, J. M. Badía-Contelles, and A. Pons-Porrata. Extended Star Clustering Algorithm. *Lecture Notes on Computer Sciences*, 2905:480–487, 2003.

[2] G. Lance and W. Williams. A general theory of classificatory sorting strategies. 1:Hierarchical systems. *Computer Journal*, 9:373–380, 1967.

[3] B. Larsen and C. Aone. Fast and Effective Text Mining Using Linear-time Document Clustering. In *KDD'99*, pages 16–22, 1999.

[4] A. Pons-Porrata, R. Berlanga-Llavori, and J. Ruiz-Shulcloper. On-line event and topic detection by using the compact sets clustering algorithm. *Journal of Intelligent and Fuzzy Systems*, 3-4:185–194, 2002.

[5] M. Steinbach, G.Karypis, and V. Kumar. A comparison of document clustering techniques. In *KDD Workshop on Text Mining*, 2000.

[6] TDT. The 2004 Topic Detection and Tracking (TDT2004) Task Definition and Evaluation Plan, version 1.0, 2004.

[7] M.-Q. Yu, W.-H. Luo, Z.-T. Zhou, and S. Bai. ICT's Approaches to HTD and Tracking at TDT2004. In *TDT2004 Workshop*, 2004.