Overview

Algorithms for hierarchical clustering: an overview

Fionn Murtagh[1,2]* and Pedro Contreras[2]

We survey agglomerative hierarchical clustering algorithms and discuss efficient implementations that are available in R and other software environments. We look at hierarchical self-organizing maps, and mixture models. We review grid-based clustering, focusing on hierarchical density-based approaches. Finally, we describe a recently developed very efficient (linear time) hierarchical clustering algorithm, which can also be viewed as a hierarchical grid-based algorithm.

How to cite this article:
WIREs Data Mining Knowl Discov 2011. doi: 10.1002/widm.53

INTRODUCTION

Agglomerative hierarchical clustering has been the dominant approach to constructing embedded classification schemes. It is our aim to direct the reader's attention to practical algorithms and methods—both efficient (from the computational and storage points of view) and effective (from the application point of view). It is often helpful to distinguish between *method*, involving a compactness criterion and the target structure of a two-way tree representing the partial order on subsets of the power set, as opposed to an *implementation*, which relates to the detail of the algorithm used.

As with many other multivariate techniques, the objects to be classified have numerical measurements on a set of variables or attributes. Hence, the analysis is carried out on the rows of an array or matrix. If we do not have a matrix of numerical values to begin with, then it may be necessary to skilfully construct such a matrix. The objects, or rows of the matrix, can be viewed as vectors in a multidimensional space (the dimensionality of this space being the number of variables or columns). A geometric framework of this type is not the only one which can be used to formulate clustering algorithms. Suitable alternative forms of storage of a rectangular array of values are not inconsistent with viewing the problem in geometric terms (and in matrix terms, e.g., expressing the adjacency relations in a graph).

Motivation for clustering in general, covering hierarchical clustering and applications, includes the following: analysis of data, interactive user interfaces, storage and retrieval, and pattern recognition.

Surveys of clustering with coverage also of hierarchical clustering include Gordon,[1] March,[2] Jain and Dubes,[3] Gordon,[4] Mirkin,[5] Jain et al.,[6] and Xu and Wunsch.[7] Lerman[8] and Janowitz[9] present overarching reviews of clustering including use of lattices that generalize trees. The case for the central role of hierarchical clustering in information retrieval was made by van Rijsbergen[10] and continued in the work of Willett and coworkers.[11] Various mathematical views of hierarchy, all expressing symmetry in one way or another, are explored by Murtagh.[12]

This paper is organized as follows. In section *Distance, Similarity, and Their Use*, we look at the issue of normalization of data, prior to inducing a hierarchy on the data. In section *Motivation*, some historical remarks and motivation are provided for hierarchical agglomerative clustering. In section *Algorithms*, we discuss the Lance–Williams formulation of a wide range of algorithms, and show how these algorithms can be expressed in graph theoretic terms and in geometric terms. In section *Efficient Hierarchical Clustering Algorithms Using Nearest Neighbor Chains*, we describe the principles of the reciprocal nearest neighbor (RNN) and nearest neighbor (NN) chain algorithm to support building a hierarchical clustering in a more efficient manner compared to the

*Correspondence to: fmurtagh@acm.org
[1]Science Foundation Ireland, Wilton Place, Dublin, Ireland
[2]Department of Computer Science, Royal Holloway, University of London, Egham, UK
DOI: 10.1002/widm.53

Lance-Williams or general geometric approaches. In section *Hierarchical Self-Organizing Maps and Hierarchical Mixture Modeling*, we overview the hierarchical Kohonen self-organizing feature map, and also hierarchical model-based clustering. We conclude this section with some reflections on divisive hierarchical clustering, in general. Section *Density- and Grid-Based Clustering Techniques* surveys developments in grid- and density-based clustering. The following section, *A New, Linear Time Grid Clustering Method: m-Adic Clustering*, presents a recent algorithm of this type, which is particularly suitable for the hierarchical clustering of massive datasets.

DISTANCE, SIMILARITY, AND THEIR USE

Before clustering comes the phase of data measurement, or measurement of the observables. Let us look at some important considerations to be taken into account. These considerations relate to the metric or other spatial embedding, comprising the first phase of the data analysis *stricto sensu*.

To group data we need a way to measure the elements and their distances relative to each other in order to decide which elements belong to a group. This can be a similarity, although on many occasions a dissimilarity measurement, or a 'stronger' distance, is used.

A distance between any pair of vectors or points i, j, k satisfies the properties of symmetry, $d(i, j) = d(j, k)$; positive definiteness, $d(i, j) > 0$ and $d(i, j) = 0$ iff $i = j$; and the triangular inequality, $d(i, j) \leq d(i, k) + d(k, j)$. If the triangular inequality is not taken into account, we have a dissimilarity. Finally, a similarity is given by $s(i, j) = \max_{i,j}\{d(i, j)\} - d(i, j)$.

When working in a vector space, a traditional way to measure distances is a Minkowski distance, which is a family of metrics defined as follows:

$$L_p(\mathbf{x}_a, \mathbf{x}_b) = \left(\sum_{i=1}^{n} |\mathbf{x}_{i,a} - \mathbf{x}_{i,b}|^p \right)^{1/p} ; \ \forall p \geq 1, \ p \in \mathbb{Z}^+, \tag{1}$$

where $\mathbb{Z}^+$ is the set of positive integers.

The Manhattan, Euclidean, and Chebyshev distances (the latter is also called maximum distance) are special cases of the Minkowski distance when $p = 1$, $p = 2$, and $p \to \infty$.

As an example of similarity, we have the *cosine* similarity, which gives the angle between two vectors. This is widely used in text retrieval to match vector queries to the dataset. The smaller the angle between a query vector and a document vector, the closer a query is to a document. The normalized cosine similarity is defined as follows:

$$s(\mathbf{x}_a, \mathbf{x}_b) = \cos(\theta) = \frac{\mathbf{x}_a \cdot \mathbf{x}_b}{\|\mathbf{x}_a\| \|\mathbf{x}_b\|}, \tag{2}$$

where $\mathbf{x}_a \cdot \mathbf{x}_b$ is the dot product and $\| \cdot \|$ is the norm.

Other relevant distances are the Hellinger, variational, Mahalanobis, and Hamming distances. Anderberg[13] gives a good review of measurement and metrics, where their interrelationships are also discussed. Also, Deza and Deza[14] have produced a comprehensive list of distances in their *Encyclopedia of Distances*.

By mapping our input data into a Euclidean space, where each object is equiweighted, we can use a Euclidean distance for the clustering that follows. Correspondence analysis is very versatile in determining a Euclidean, factor space from a wide range of input data types, including frequency counts, mixed qualitative and quantitative data values, ranks or scores, and others. Further reading on this is to be found in Benzécri[15] and Le Roux and Rouanet,[16] and Murtagh.[17]

AGGLOMERATIVE HIERARCHICAL CLUSTERING

Motivation

Agglomerative hierarchical clustering algorithms can be characterized as *greedy*, in the algorithmic sense. A sequence of irreversible algorithm steps is used to construct the desired data structure. Assume that a pair of clusters, including possibly singletons, is merged or agglomerated at each step of the algorithm. Then the following are equivalent views of the same output structure constructed on n objects: a set of $n - 1$ partitions, starting with the fine partition consisting of n classes and ending with the trivial partition consisting of just one class, the entire object set; a binary tree (one or two child nodes at each nonterminal node) commonly referred to as a dendrogram; a partially ordered set (poset) which is a subset of the power set of the n objects; and an ultrametric topology on the n objects.

An ultrametric, or tree metric, defines a stronger topology compared to, e.g., a Euclidean metric geometry. For three points, i, j, k, metric and ultrametric respect the properties of symmetry (d, $d(i, j) = d(j, i)$) and positive definiteness ($d(i, j) > 0$ and if $d(i, j) = 0$ then $i = j$). A metric though (as noted in section *Distance, Similarity, and Their Use*) satisfies the triangular inequality, $d(i, j) \leq d(i, k) + d(k, j)$ while

an ultrametric satisfies the strong triangular or ultrametric (or non-Archimedean), inequality, $d(i, j) \leq \max\{d(i, k), d(k, j)\}$. In section *Distance, Similarity, and Their Use*, there was further discussion on metrics.

The single linkage hierarchical clustering approach outputs a set of clusters (to use graph theoretic terminology, a set of maximal connected subgraphs) at each level—or for each threshold value which produces a new partition. The single linkage method with which we begin is one of the oldest methods, its origins being traced to Polish researchers in the 1950s.[18] The name *single linkage* arises as the interconnecting dissimilarity between two clusters or components is defined as the least interconnecting dissimilarity between a member of one and a member of the other. Other hierarchical clustering methods are characterized by other functions of the interconnecting linkage dissimilarities.

As early as the 1970s, it was held that about 75% of all published work on clustering employed hierarchical algorithms.[19] Interpretation of the information contained in a dendrogram is often of one or more of the following kinds: set inclusion relationships, partition of the object sets, and significant clusters.

Much early work on hierarchical clustering was in the field of biological taxonomy, from the 1950s and more so from the 1960s onward. The central reference in this area, the first edition of which dates from the early 1960s, is Ref 20. One major interpretation of hierarchies has been the evolution relationships between the organisms under study. It is hoped, in this context, that a dendrogram provides a sufficiently accurate model of underlying evolutionary progression.

A common interpretation made of hierarchical clustering is to derive a partition. A further type of interpretation is instead to detect maximal (i.e., disjoint) clusters of interest at varying levels of the hierarchy. Such an approach is used by Rapoport and Fillenbaum[21] in a clustering of colors based on semantic attributes. Lerman[8] developed an approach for finding significant clusters at varying levels of a hierarchy, which has been widely applied. By developing a wavelet transform *on* a dendrogram,[22] which amounts to a wavelet transform in the associated ultrametric topological space, the most important—in the sense of best approximating—clusters can be determined. Such an approach is a topological one (i.e., based on sets and their properties) as contrasted with more widely used optimization or statistical approaches.

In summary, a dendrogram collects together many of the proximity and classificatory relationships in a body of data. It is a convenient representation which answers such questions as: 'How many useful groups are in this data?' and 'What are the salient interrelationships present?' But it can be noted that differing answers can feasibly be provided by a dendrogram for most of these questions, depending on the application.

Algorithms

A wide range of agglomerative hierarchical clustering algorithms have been proposed at one time or another. Such hierarchical algorithms may be conveniently broken down into two groups of methods. The first group is that of linkage methods—the single, complete, weighted, and unweighted average linkage methods. These are methods for which a graph representation can be used. Sneath and Sokal[20] may be consulted for many other graph representations of the stages in the construction of hierarchical clusterings.

The second group of hierarchical clustering methods are methods which allow the cluster centers to be specified (as an average or a weighted average of the member vectors of the cluster). These methods include the centroid, median, and minimum variance methods.

The latter may be specified either in terms of dissimilarities, alone, or alternatively in terms of cluster center coordinates and dissimilarities. A very convenient formulation, in dissimilarity terms, which embraces all the hierarchical methods mentioned so far, is the *Lance–Williams dissimilarity update formula*. If points (objects) i and j are agglomerated into cluster $i \cup j$, then we must simply specify the new dissimilarity between the cluster and all other points (objects or clusters). The formula is

$$d(i \cup j, k) = \alpha_i d(i, k) + \alpha_j d(j, k) + \beta d(i, j) + \gamma \mid d(i, k) - d(j, k) \mid,$$

where α_i, α_j, β, and γ define the agglomerative criterion. Values of these are listed in the second column of Table 1. In the case of the single link method, using $\alpha_i = \alpha_j = \frac{1}{2}$, $\beta = 0$, and $\gamma = -\frac{1}{2}$ gives us

$$d(i \cup j, k) = \frac{1}{2} d(i, k) + \frac{1}{2} d(j, k) - \frac{1}{2} \mid d(i, k) - d(j, k) \mid,$$

which, it may be verified, can be rewritten as

$$d(i \cup j, k) = \min\{d(i, k), d(j, k)\}.$$

TABLE 1 | Specifications of Seven Hierarchical Clustering Methods

Hierarchical Clustering Methods (and Aliases)	Lance–Williams Dissimilarity Update Formula	Coordinates of Center of Cluster, which Agglomerates Clusters i and j	Dissimilarity between Cluster Centers g_i and g_j
Single link (nearest neighbor)	$\alpha_i = 0.5$ $\beta = 0$ $\gamma = -0.5$ (More simply: $\min\{d_{ik}, d_{jk}\}$)		
Complete link (diameter)	$\alpha_i = 0.5$ $\beta = 0$ $\gamma = 0.5$ (More simply: $\max\{d_{ik}, d_{jk}\}$)		
Group average (average link, UPGMA)	$\alpha_i = \frac{\mid i \mid}{\mid i \mid + \mid j \mid}$ $\beta = 0$ $\gamma = 0$		
McQuitty's method (WPGMA)	$\alpha_i = 0.5$ $\beta = 0$ $\gamma = 0$		
Median method (Gower's, WPGMC)	$\alpha_i = 0.5$ $\beta = -0.25$ $\gamma = 0$	$\mathbf{g} = \frac{\mathbf{g}_i + \mathbf{g}_j}{2}$	$\lVert \mathbf{g}_i - \mathbf{g}_j \rVert^2$
Centroid (UPGMC)	$\alpha_i = \frac{\mid i \mid}{\mid i \mid + \mid j \mid}$ $\beta = -\frac{\mid i \mid \mid j \mid}{(\mid i \mid + \mid j \mid)^2}$ $\gamma = 0$	$\mathbf{g} = \frac{\mid i \mid \mathbf{g}_i + \mid j \mid \mathbf{g}_j}{\mid i \mid + \mid j \mid}$	$\lVert \mathbf{g}_i - \mathbf{g}_j \rVert^2$
Ward's method (minimum variance, error sum of squares)	$\alpha_i = \frac{\mid i \mid + \mid k \mid}{\mid i \mid + \mid j \mid + \mid k \mid}$ $\beta = -\frac{\mid k \mid}{\mid i \mid + \mid j \mid + \mid k \mid}$ $\gamma = 0$	$\mathbf{g} = \frac{\mid i \mid \mathbf{g}_i + \mid j \mid \mathbf{g}_j}{\mid i \mid + \mid j \mid}$	$\frac{\mid i \mid \mid j \mid}{\mid i \mid + \mid j \mid} \lVert \mathbf{g}_i - \mathbf{g}_j \rVert^2$

$|i|$ is the number of objects in cluster i; $\mathbf{g}_i$ is a vector in m-space (m is the set of attributes), either an initial point or a cluster center; $\|.\|$ is the norm in the Euclidean metric. The names UPGMA, etc. are because of Sneath and Sokal.[20] Coefficient α_j, with index j, is defined identically to coefficient α_i with index i. Finally, the Lance and Williams recurrence formula is (with $|.|$ expressing absolute value)

$$d_{i \cup j,k} = \alpha_i d_{ik} + \alpha_j d_{jk} + \beta d_{ij} + \gamma \mid d_{ik} - d_{jk} \mid .$$

Using other update formulas, as given in column 2 of Table 1, allows the other agglomerative methods to be implemented in a very similar way to the implementation of the single link method.

In the case of the methods which use cluster centers, we have the center coordinates (in column 3 of Table 1) and dissimilarities as defined between cluster centers (column 4 of Table 1). The Euclidean distance must be used for equivalence between the two approaches. In the case of the *median method*, for instance, we have the following (cf. Table 1).

Let $\mathbf{a}$ and $\mathbf{b}$ be two points (i.e., m-dimensional vectors: these are objects or cluster centers) which have been agglomerated, and let $\mathbf{c}$ be another point. From the Lance–Williams dissimilarity update formula, using squared Euclidean distances, we have

$$\begin{aligned} d^2(a \cup b, c) &= \frac{d^2(a,c)}{2} + \frac{d^2(b,c)}{2} - \frac{d^2(a,b)}{4} \\ &= \frac{\lVert \mathbf{a} - \mathbf{c} \rVert^2}{2} + \frac{\lVert \mathbf{b} - \mathbf{c} \rVert^2}{2} - \frac{\lVert \mathbf{a} - \mathbf{b} \rVert^2}{4}. \end{aligned} \quad (3)$$

The new cluster center is $(\mathbf{a} + \mathbf{b})/2$, so that its distance to point $\mathbf{c}$ is

$$\left\lVert \mathbf{c} - \frac{\mathbf{a} + \mathbf{b}}{2} \right\rVert^2. \quad (4)$$

That these two expressions are identical is readily verified. The correspondence between these two perspectives on the one agglomerative criterion is

BOX 1: STORED DATA APPROACH

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

Step 2: Replace two points clustered by representative point (center of gravity) or by cluster fragment.

Step 3: Return to step 1, treating clusters as well as remaining objects, until all objects are in one cluster.

similarly proved for the centroid and minimum variance methods. This is an example of a "stored data" algorithm (see Box 1).

For cluster center methods, and with suitable alterations for graph methods, the following algorithm is an alternative to the general dissimilarity-based algorithm. The latter may be described as a "stored dissimilarities approach".[13]

In steps 1 and 2, "point" refers either to objects or clusters, both of which are defined as vectors in the case of cluster center methods. This algorithm is justified by storage considerations because we have $O(n)$ storage required for n initial objects and $O(n)$ storage for the $n-1$ (at most) clusters. In the case of linkage methods, the term "fragment" in step 2 refers (in the terminology of graph theory) to a connected component in the case of the single link method and to a clique or complete subgraph in the case of the complete link method. Without consideration of any special algorithmic "speed-ups", the overall complexity of the above algorithm is $O(n^3)$ due to the repeated calculation of dissimilarities in step 1, coupled with $O(n)$ iterations through steps 1, 2 and 3. Although the stored data algorithm is instructive, it does not lend itself to efficient implementations. In the next section, we look at the RNN and mutual NN algorithms which can be used in practice for implementing agglomerative hierarchical clustering algorithms.

Before concluding this overview of agglomerative hierarchical clustering algorithms, we will describe briefly the minimum variance method.

The variance or spread of a set of points (i.e., the average of the sum of squared distances from the center) has been a point of departure for specifying clustering algorithms. Many of these algorithms—iterative, optimization algorithms as well as the hierarchical, agglomerative algorithms—are described and appraised in Ref 23. The use of variance in a clustering criterion links the resulting clustering to other data-analytic techniques which involve a decomposition of variance, and make the minimum variance agglomerative strategy particularly suitable for synoptic clustering. Hierarchies are also more balanced with this agglomerative criterion, which is often of practical advantage.

The minimum variance method produces clusters which satisfy compactness and isolation criteria. These criteria are incorporated into the dissimilarity. We seek to agglomerate two clusters, c_1 and c_2, into cluster c such that the within-class variance of the partition thereby obtained is minimum. Alternatively, the between-class variance of the partition obtained is to be maximized. Let P and Q be the partitions prior to, and subsequent to, the agglomeration; let $p_1, p_2, \ldots$ be classes of the partitions:

$$P = \{p_1, p_2, \ldots, p_k, c_1, c_2\},$$
$$Q = \{p_1, p_2, \ldots, p_k, c\}.$$

Letting V denote *variance*, then in agglomerating two classes of P, the variance of the resulting partition [i.e., $V(Q)$] will necessarily decrease: therefore in seeking to minimize this decrease, we simultaneously achieve a partition with maximum between-class variance. The criterion to be optimized can then be shown to be

$$V(P) - V(Q) = V(c) - V(c_1) - V(c_2) = \frac{|\, c_1 \,|\, |\, c_2 \,|}{|\, c_1 \,| + |\, c_2 \,|} \|\mathbf{c}_1 - \mathbf{c}_2\|^2,$$

which is the dissimilarity given in Table 1. This is a dissimilarity which may be determined for any pair of classes of partition P; and the agglomerands are those classes, c_1 and c_2, for which it is minimum.

It may be noted that if c_1 and c_2 are singleton classes, then $V(\{c_1, c_2\}) = \frac{1}{2}\|\mathbf{c}_1 - \mathbf{c}_2\|^2$, i.e., the variance of a pair of objects is equal to half their Euclidean distance.

EFFICIENT HIERARCHICAL CLUSTERING ALGORITHMS USING NEAREST NEIGHBOR CHAINS

Early, efficient algorithms for hierarchical clustering are because of Sibson,[24] Rohlf,[25] and Defays.[26] Their $O(n^2)$ implementations of the single link method and of a (nonunique) complete link method, respectively, have been widely cited.

In the early 1980s, a range of significant improvements[27,28] were made to the Lance–Williams, or related, dissimilarity update schema, which had been in wide use since the mid-1960s. Murtagh[29,30] presents a survey of these algorithmic improvements. We will briefly describe them here. The new

FIGURE 1 | Five points, showing nearest neighbors and reciprocal nearest neighbors.

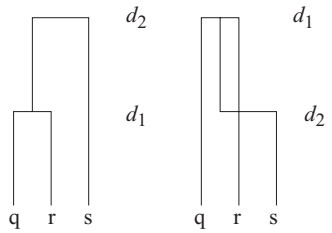

FIGURE 2 | Alternative representations of a hierarchy with an inversion. Assuming dissimilarities, as we go vertically up, agglomerative criterion values (d_1, d_2) increase so that $d_2 > d_1$. But here, undesirably, $d_2 < d_1$ and the "crossover" or inversion (right panel) arises.

algorithms, which have the potential for *exactly* replicating results found in the classical but more computationally expensive way, are based on the construction of *nearest neighbor chains* and *reciprocal* or mutual NNs.

An NN chain consists of an arbitrary point (a in Figure 1), followed by its NN (b in Figure 1), followed by the NN from among the remaining points (c, d, and e in Figure 1) of this second point; and so on until we necessarily have some pair of points which can be termed reciprocal or mutual NNs. (Such a pair of RNNs may be the first two points in the chain; and we have assumed that no two dissimilarities are equal.)

In constructing an NN chain, irrespective of the starting point, we may agglomerate a pair of RNNs as soon as they are found. What guarantees that we can arrive at the same hierarchy as if we used traditional "stored dissimilarities" or "stored data" algorithms? Essentially, this is the same condition as that under which no inversions or reversals are produced by the clustering method. Figure 2 gives an example of this, where s is agglomerated at a lower criterion value (i.e., dissimilarity) than was the case at the previous agglomeration between q and r. Our ambient space has thus contracted because of the agglomeration. This is because of the algorithm used—in particular, the agglomeration criterion—and it is something we would normally wish to avoid.

This is formulated as

$$\text{Inversion impossible if: } d(i, j) < d(i, k) \text{ or } d(j, k) \geq d(i, j) < d(i \cup j, k).$$

BOX 2: NEAREST NEIGHBOR CHAIN ALGORITHM

Step 1: Select a point arbitrarily.

Step 2: Grow the nearest neighbor (NN) chain from this point until a pair of reciprocal nearest neighbors (RNNs) is obtained.

Step 3: Agglomerate these points (replacing with a cluster point, or updating the dissimilarity matrix).

Step 4: From the point which preceded the RNNs (or from any other arbitrary point if the first two points chosen in steps 1 and 2 constituted a pair of RNNs), return to step 2 until only one point remains.

This is one form of Bruynooghe's *reducibility property*.[31,32] Using the Lance–Williams dissimilarity update formula, it can be shown that the minimum variance method does not give rise to inversions; neither do the linkage methods; but the median and centroid methods cannot be guaranteed *not* to have inversions.

To return to Figure 1, if we are dealing with a clustering criterion which precludes inversions, then c and d can justifiably be agglomerated because no other point (e.g., b or e) could have been agglomerated to either of these.

The processing required, following an agglomeration, is to update the NNs of points such as b in Figure 1 (and on account of such points, this algorithm was dubbed *algorithme des célibataires*, or bachelors' algorithm, in Ref 27). Box 2 gives a summary of the algorithm:

In Murtagh[29,30,32] and Day and Edelsbrunner,[33] one finds discussions of $O(n^2)$ time and $O(n)$ space implementations of Ward's minimum variance (or error sum of squares) method and of the centroid and median methods. The latter two methods are termed the UPGMC and WPGMC criteria by Sneath and Sokal.[20] Now, a problem with the cluster criteria used by these latter two methods is that the reducibility property is not satisfied by them. This means that the hierarchy constructed may not be unique as a result of inversions or reversals (nonmonotonic variation) in the clustering criterion value determined in the sequence of agglomerations.

Murtagh[29,30] describes $O(n^2)$ time and $O(n^2)$ space implementations for the single link method, the complete link method and for the weighted and unweighted group average methods (WPGMA and UPGMA). This approach is quite general vis-à-vis the

dissimilarity used and can also be used for hierarchical clustering methods other than those mentioned.

Day and Edelsbrunner[33] prove the exact $O(n^2)$ time complexity of the centroid and median methods using an argument related to the combinatorial problem of optimally packing hyperspheres into an m-dimensional volume. They also address the question of metrics: results are valid in a wide class of distances including those associated with the Minkowski metrics.

The construction and maintenance of the NN chain as well as the carrying out of agglomerations whenever reciprocal NNs meet, both offer possibilities for distributed implementation. Implementations on a parallel machine architecture were described by Willett.[34]

Evidently (from Table 1), both coordinate data and graph (e.g., dissimilarity) data can be input to these agglomerative methods. Gillet et al.[35] in the context of clustering chemical structure databases refer to the common use of the Ward method, based on the RNN algorithm, on datasets of a few hundred thousand molecules.

Applications of hierarchical clustering to bibliographic information retrieval are assessed by Griffiths et al.[11] Ward's minimum variance criterion is favored.

From details in Ref 36, the Institute of Scientific Information (ISI) clusters citations (science, and social science) by first clustering highly cited documents based on a single linkage criterion, and then four more passes are made through the data to create a subset of a single linkage hierarchical clustering.

In the CLUSTAN and R statistical data analysis packages [in addition to hclust in R, see flashClust due to P. Langfelder and available on CRAN (Comprehensive R Archive Network) at: http://cran.r-project.org], there are implementations of the NN-chain algorithm for the minimum variance agglomerative criterion. A property of the minimum variance agglomerative hierarchical clustering method is that we can use weights on the objects on which we will induce a hierarchy. By default, these weights are identical and equal to 1. Such weighting of observations to be clustered is an important and practical aspect of these software packages.

HIERARCHICAL SELF-ORGANIZING MAPS AND HIERARCHICAL MIXTURE MODELING

Visual metaphors have always influenced hierarchical clustering. A family tree, e.g., is a natural enough way to introduce informally the notion of a metric on a tree, i.e., an ultrametric. A mathematical graph gives rise to a useful visual metaphor (cf. how typically a transport system is displayed) and the early tome on clustering, including discussion on hierarchical clustering,[20] is amply illustrated using graphs. In this section, we look at how spatial display has been used for hierarchical clustering. This combines in an intuitive way both visualization and data analysis.

It is quite impressive how two-dimensional (2D or, for that matter, 3D) image signals can handle with ease the scalability limitations of clustering and many other data processing operations. The contiguity imposed on adjacent pixels or grid cells bypasses the need for NN finding. It is very interesting therefore to consider the feasibility of taking problems of clustering massive datasets into the 2D image domain. The Kohonen self-organizing feature map exemplifes this well. In its basic variant,[37,38] it can be formulated in terms of k-means clustering subject to a set of interrelationships between the cluster centers.[39]

Kohonen maps lend themselves well for hierarchical representation. Lampinen and Oja,[40] Dittenbach et al.,[41] and Endo et al.[42] elaborate on the Kohonen map in this way. An example application in character recognition is Ref 43.

A short, informative review of hierarchical self-organizing maps is provided by Vicente and Vellido.[44] These authors also review what they term as probabilistic hierarchical models. This includes putting into a hierarchical framework the following: Gaussian mixture models, and a probabilistic—Bayesian—alternative to the Kohonen self-organizing map termed generative topographic mapping (GTM).

GTM can be traced to the Kohonen self-organizing map in the following way. First, we consider the hierarchical map as brought about through a growing process, i.e., the target map is allowed to grow in terms of layers, and of grid points within those layers. Second, we impose an explicit probability density model on the data. Tino and Nabney[45] discuss how the local hierarchical models are organized in a hierarchical way.

In Wang et al.[46] an alternating Gaussian mixture modeling, and principal component analysis, is described, in this way furnishing a hierarchy of model-based clusters. Akaike information criterion (AIC), is used for selection of the best cluster model overall.

Murtagh et al.[47] use a top level Gaussian mixture modeling with the (spatially aware) PLIC, pseudo-likelihood information criterion, used for cluster selection and identifiability. Then at the next level—and potentially also for further divisive, hierarchical levels—the Gaussian mixture modeling

is continued but now using the marginal distributions within each cluster, and using the analogous Bayesian clustering identifiability criterion which is the Bayesian information criterion, (BIC). The resulting output is referred to as a model-based cluster tree.

The model-based cluster tree algorithm of Murtagh et al.[47] is a divisive hierarchical algorithm. Earlier in this article, we considered agglomerative algorithms. However, it is often feasible to implement a divisive algorithm instead, especially when a graph cut (for example) is important for the application concerned. Mirkin[5] describes divisive Ward, minimum variance hierarchical clustering, which also is closely related to a bisecting *k*-means.

A class of methods under the name of spectral clustering uses eigenvalue/eigenvector reduction on the (graph) adjacency matrix. As von Luxburg[48] points out in reviewing this field of spectral clustering, such methods have "been discovered, rediscovered, and extended many times in different communities". Far from seeing this great deal of work on clustering in any sense in a pessimistic way, we see the perennial and pervasive interest in clustering as testifying to the continual renewal and innovation in algorithm developments, faced with application needs.

It is indeed interesting to note how the clusters in a hierarchical clustering may be *defined* by the eigenvectors of a dissimilarity matrix, but subject to carrying out the eigenvector reduction in a particular algebraic structure, a semi-ring with additive and multiplicative operations given by 'min' and 'max', respectively.[49]

In section *Density and Grid-Based Clustering Techniques*, the themes of mapping, and of divisive algorithm, are frequently taken in a somewhat different direction. As always, the application at issue is highly relevant for the choice of the hierarchical clustering algorithm.

DENSITY- AND GRID-BASED CLUSTERING TECHNIQUES

Many modern clustering techniques focus on large datasets. In Xu and Wunsch[50] these are classified as follows:

- Random sampling,
- Data condensation,
- Density-based approaches,
- Grid-based approaches,
- Divide and conquer,
- Incremental learning.

From the point of view of this paper, we select density- and grid-based approaches, i.e., methods that either look for data densities or split the data space into cells when looking for groups. In this section, we take a look at these two families of methods.

The main idea is to use a grid-like structure to split the information space, separating the dense grid regions from the less dense ones to form groups.

In general, a typical approach within this category will consist of the following steps as presented by Grabusts and Borisov[51]:

1. Creating a grid structure, i.e., partitioning the data space into a finite number of nonoverlapping cells;
2. Calculating the cell density for each cell;
3. Sorting of the cells according to their densities;
4. Identifying cluster centers;
5. Traversal of neighbor cells.

Some of the more important algorithms within this category are the following:

- *STING:* STatistical INformation Grid-based clustering was proposed by Wang et al.[52] who divide the spatial area into rectangular cells represented by a hierarchical structure. The root is at hierarchical level 1, its children at level 2, and so on. This algorithm has a computational complexity of $O(K)$, where K is the number of cells in the bottom layer. This implies that scaling this method to higher dimensional spaces is difficult.[53] For example, if in high-dimensional data space each cell has four children, then the number of cells in the second level will be 2^m, where m is the dimensionality of the database.
- *OptiGrid:* Optimal Grid-Clustering was introduced by Hinneburg and Keim[53] as an efficient algorithm to cluster high-dimensional databases with noise. It uses data partitioning based on divisive recursion by multidimensional grids, focusing on separation of clusters by hyperplanes. A cutting plane is chosen which goes through the point of minimal density, therefore splitting two dense half-spaces. This process is applied recursively with each subset of data. This algorithm is hierarchical, with time complexity of $O(n \cdot m)$.[54]
- *GRIDCLUS:* Proposed by Schikute,[55] GRIDCLUS is a hierarchical algorithm for clustering very large datasets. It uses a

multidimensional data grid to organize the space surrounding the data values rather than organize the data themselves. Thereafter, patterns are organized into blocks, which in turn are clustered by a topological neighbor search algorithm. Five main steps are involved in the GRIDCLUS method: (1) insertion of points into the grid structure, (2) calculation of density indices, (3) sorting the blocks with respect to their density indices, (4) identification of cluster centers, and (5) traversal of neighbor blocks.

- *WaveCluster:* This clustering technique, proposed by Sheikholeslami et al.,[56] defines a uniform two-dimensional grid on the data and represents the data points in each cell by the number of points. Thus the data points become a set of gray-scale points, which is treated as an image. Then the problem of looking for clusters is transformed into an image segmentation problem, where wavelets are used to take advantage of their multiscaling and noise reduction properties. The basic algorithm is as follows: (1) create a data grid and assign each data object to a cell in the grid, (2) apply the wavelet transform to the data, (3) use the average subimage to find connected clusters (i.e., connected pixels), and (4) map the resulting clusters back to the points in the original space. There is also a great deal of other work that is based on using the wavelet and other multiresolution transforms for segmentation.

Further grid-based clustering algorithms can be found in the following: Chang and Jin,[57] Park and Lee,[58] Gan et al.,[54] and Xu and Wunsch.[50]

Density-based clustering algorithms are defined as dense regions of points, which are separated by low-density regions. Therefore, clusters can have an arbitrary shape and the points in the clusters may be arbitrarily distributed. An important advantage of this methodology is that only one scan of the dataset is needed and it can handle noise effectively. Furthermore, the number of clusters to initialize the algorithm is not required.

Some of the more important algorithms in this category include the following:

- *DBSCAN:* Density-Based Spatial Clustering of Applications with Noise was proposed by Ester et al.[59] to discover arbitrarily shaped clusters. As it finds clusters based on density it does not need to know the number of clusters at initialization time. This algorithm has been widely used and has many variations (e.g., see GDBSCAN by Sander et al.,[60] PDBSCAN by Xu et al.,[61] and DBCluC by Zaïane and Lee.[62]
- *BRIDGE:* Proposed by Dash et al.,[63] it uses a hybrid approach integrating k-means to partition the dataset into k clusters, and then density-based algorithm DBSCAN is applied to each partition to find dense clusters.
- *DBCLASD:* Distribution-Based Clustering of LArge Spatial Databases (see Ref 64) assumes that data points within a cluster are uniformly distributed. The cluster produced is defined in terms of the NN distance.
- *DENCLUE:* DENsity-based CLUstering aims to cluster large multimedia data. It can find arbitrarily shaped clusters and at the same time deals with noise in the data. This algorithm has two steps. First, a precluster map is generated, and the data is divided in hypercubes where only the populated are considered. The second step takes the highly populated cubes and cubes that are connected to a highly populated cube to produce the clusters. For a detailed presentation of these steps, see Hinneburg and Keim.[65]
- *CUBN:* This has three steps. First, an erosion operation is carried out to find border points. Second, the NN method is used to cluster the border points. Finally, the NN method is used to cluster the inner points. This algorithm is capable of finding non-spherical shapes and wide variations in size. Its computational complexity is $O(n)$ with n being the size of the dataset. For a detailed presentation of this algorithm, see Ref 66.

A NEW, LINEAR TIME GRID CLUSTERING METHOD: m-ADIC CLUSTERING

Algorithms described earlier in this paper, in sections Efficient Hierarchical Clustering Algorithms Using Nearest Neighbor Chains and Agglomerative Hierarchical Clustering, are of quadratic worst case computational time. Alternatively expressed, they are of computational complexity $O(n^2)$ for n observations. This becomes quite impractical for datasets of any realistic size. In this section, we describe a recent development that allows a hierarchical clustering to be constructed in (worst case) linear time.

In the previous section, we have seen a number of clustering methods that split the data space into cells, cubes, or dense regions to locate high density areas that can be further studied to find clusters.

For large datasets, clustering via an m-adic (m integer, which if a prime is usually denoted as p) expansion is possible, with the advantage of doing so in linear time for the clustering algorithm based on this expansion. The usual base 10 system for numbers is none other than the case of $m = 10$ and the base 2 or binary system can be referred to as 2-adic, where $p = 2$. Let us consider the following distance relating to the case of vectors x and y with 1 attribute, hence unidimensional:

$$d_{\mathcal{B}}(x, y) = \begin{cases} 1 & \text{if } x_1 \neq y_1, \\ \inf m^{-k} & x_k = y_k \; 1 \leq k \leq |K|. \end{cases} \quad (5)$$

This distance defines the longest common prefix of strings. A space of strings, with this distance, is a Baire space. Thus we call this the Baire distance: here the longer the common prefix, the closer a pair of sequences. What is of interest to us here is this longest common prefix metric, which is an ultrametric.[67]

For example, let us consider two such values, x and y. We take x and y to be bounded by 0 and 1. Each are of some precision, and we take the integer $|K|$ to be the maximum precision.

Thus we consider ordered sets x_k and y_k for $k \in K$. So, $k = 1$ is the index of the first decimal place of precision; $k = 2$ is the index of the second decimal place; ...; $k = |K|$ is the index of the $|K|$th decimal place. The cardinality of the set K is the precision with which a number, x, is measured.

Consider as examples $x_k = 0.478$; and $y_k = 0.472$. In these cases, $|K| = 3$. Start from the first decimal position. For $k = 1$, we have $x_k = y_k = 4$. For $k = 2$, $x_k = y_k$. But for $k = 3$, $x_k \neq y_k$. Hence, their Baire distance is 10^{-2} for base $m = 10$.

It is seen that this distance splits a unidimensional string of decimal values into a 10-way hierarchy, in which each leaf can be seen as a grid cell. From Eq. (5), we can read off the distance between points assigned to the same grid cell. All pairwise distances of points assigned to the same cell are the same.

Clustering, using this Baire distance, has been applied to large datasets in areas such as chemoinformatics,[67] astronomy,[68] and text retrieval.[69]

CONCLUSIONS

Hierarchical clustering methods, with roots going back to the 1960s and 1970s, are continually replenished with new challenges. As a family of algorithms, they are central to the addressing of many important problems. Their deployment in many application domains testifies to how hierarchical clustering methods will remain crucial for a long time to come.

We have looked at both traditional agglomerative hierarchical clustering, and more recent developments in grid or cell-based approaches. We have discussed various algorithmic aspects, including well-definedness (e.g., inversions) and computational properties. We have also touched on a number of application domains, again in areas that reach back over some decades (chemoinformatics) or many decades (information retrieval, which motivated much early work in clustering, including hierarchical clustering), and more recent application domains (such as hierarchical model-based clustering approaches).

REFERENCES

1. Gordon AD. *Classification*. London: Chapman and Hall; 1981.
2. March ST. Techniques for structuring database records. *ACM Comput Surv* 1983, 15:45–79.
3. Jain AK, Dubes RC. *Algorithms For Clustering Data*. Englwood Cliffs, NJ: Prentice-Hall; 1988.
4. Gordon AD. A review of hierarchical classification. *J R Stat Soc A* 1987, 150:119–137.
5. Mirkin B. *Mathematical Classification and Clustering*. Dordrecht, The Netherlands: Kluwer; 1996.
6. Jain AK, Murty, MN, Flynn PJ. Data clustering: a review. *ACM Comput Surv* 1999, 31:264–323.
7. Xu R, Wunsch D. Survey of clustering algorithms. *IEEE Trans Neural Netw* 2005, 16:645–678.
8. Lerman IC. *Classification et Analyse Ordinale des Données*. Paris: Dunod; 1981.
9. Janowitz MF. *Ordinal and Relational Clustering*. Singapore: World Scientific; 2010.
10. van Rijsbergen CJ. *Information Retrieval*. 2nd ed. London: Butterworths; 1979.
11. Griffiths A, Robinson LA, Willett P. Hierarchic agglomerative clustering methods for automatic document classification. *J Doc* 1984, 40:175–205.
12. Murtagh F. Symmetry in data mining and analysis: a unifying view based on hierarchy. *Proc Steklov Inst Math* 2009, 265:177–198.
13. Anderberg MR. *Cluster Analysis for Applications*. New York: Academic Press; 1973.

14. Deza MM, Deza E. *Encyclopedia of Distances*. Berlin: Springer; 2009.
15. Benzécri JP. *L'Analyse des Données. I. La Taxinomie*. 3rd ed. Paris: Dunod; 1979.
16. Le Roux B, Rouanet H. *Geometric Data Analysis: From Correspondence Analysis to Structured Data Analysis*. Dordrecht: Kluwer; 2004.
17. Murtagh F. *Correspondence Analysis and Data Coding with Java and R*. Boca Raton, FL: Chapman and Hall; 2005.
18. Graham RH, Hell P. On the history of the minimum spanning tree problem. *Ann Hist Comput* 1985, 7:43–57.
19. Blashfield RK, Aldenderfer MS. The literature on cluster analysis *Multivariate Behav Res* 1978, 13:271–295.
20. Sneath PHA, Sokal RR. *Numerical Taxonomy*. San Francisco, CA: Freeman; 1973.
21. Rapoport A, Fillenbaum S. An experimental study of semantic structures. In: Romney AK, Shepard RN, Nerlove SB, eds. *Multidimensional Scaling: Theory and Applications in the Behavioral Sciences. Vol. 2, Applications*. New York: Seminar Press; 1972, 93–131.
22. Murtagh F. The Haar wavelet transform of a dendrogram. *J Classif* 2007, 24:3–32.
23. Wishart D. Mode analysis: a generalization of nearest neighbour which reduces chaining effects. In: Cole AJ, ed. *Numerical Taxonomy*. New York: Academic Press; 1969, 282–311.
24. Sibson R. SLINK: an optimally efficient algorithm for the single link cluster method. *Comput J* 1973, 16:30–34.
25. Rohlf FJ. Algorithm 76: hierarchical clustering using the minimum spanning tree. *Comput J* 1973, 16:93–95.
26. Defays D. An efficient algorithm for a complete link method. *Comput J* 1977, 20:364–366.
27. de Rham C. La classification hiérarchique ascendante selon la méthode des voisins réciproques. *Les Cahiers de l'Analyse des Données* 1980, V:135–144.
28. Juan J. Programme de classification hiérarchique par l'algorithme de la recherche en chaîne des voisins réciproques. *Les Cahiers de l'Analyse des Données* 1982, VII:219–225.
29. Murtagh F. A survey of recent advances in hierarchical clustering algorithms. *Comput J* 1983, 26:354–359.
30. Murtagh F. *Multidimensional Clustering Algorithms*. Würzburg, Germany: Physica-Verlag; 1985.
31. Bruynooghe M. Méthodes nouvelles en classification automatique des données taxinomiques nombreuses. *Statistique et Analyse des Données* 1977, 3:24–42.
32. Murtagh F. Complexities of hierarchic clustering algorithms: state of the art. *Comput Stat Quart* 1984, 1:101–113.
33. Day WHE, Edelsbrunner H. Efficient algorithms for agglomerative hierarchical clustering methods. *J Classif* 1984, 1:7–24.
34. Willett P. Efficiency of hierarchic agglomerative clustering using the ICL distributed array processor. *J Doc* 1989, 45:1–45.
35. Gillet VJ, Wild DJ, Willett P, Bradshaw J. Similarity and dissimilarity methods for processing chemical structure databases. *Comput J* 1998, 41:547–558.
36. White HD, McCain KW. Visualization of literatures. *Annu Rev Inform Sci Technol* 1997, 32:99–168.
37. Kohonen T. *Self-Organization and Associative Memory*. Berlin: Springer; 1984.
38. Kohonen T. *Self-Organizing Maps*. 3rd ed. Berlin: Springer; 2001.
39. Murtagh F, Hernández-Pajares M. The Kohonen self-organizing map method: an assessment. *J Classif* 1995, 12:165–190.
40. Lampinen J, Oja E. Clustering properties of hierarchical self-organizing maps. *J Math Imaging Vision* 1992, 2:261–272.
41. Dittenbach M, Rauber A, Merkl D. Uncovering the hierarchical structure in data using the growing hierarchical self-organizing map. *Neurocomputing*, 2002, 48:199–216.
42. Endo M, Ueno M, Tanabe T. A clustering method using hierarchical self-organizing maps. *J VLSI Signal Process* 2002, 32:105–118.
43. Miikkulainien R. Script recognition with hierarchical feature maps. *Connect Sci* 1990, 2:83–101.
44. Vicente D, Vellido A. Review of hierarchical models for data clustering and visualization. In: Giráldez R, Riquelme JC, Aguilar-Ruiz JS, eds. *Tendencias de la Minería de Datos en España*. Seville, Spain: Red Española de Minería de Datos; 2004.
45. Tino P, Nabney I. Hierarchical GTM: constructing localized non-linear projection manifolds in a principled way. *IEEE Trans Pattern Anal Mach Intell* 2002, 24:639–656.
46. Wang Y, Freedman MI, Kung S-Y. Probabilistic principal component subspaces: a hierarchical finite mixture model for data visualization. *IEEE Trans Neural Netw* 2000, 11:625–636.
47. Murtagh F, Raftery AE, Starck JL. Bayesian inference for multiband image segmentation via model-based clustering trees. *Image Vision Comput* 2005, 23:587–596.
48. von Luxburg U. A tutorial on spectral clustering. *Stat Comput* 1997, 17:395–416.
49. Gondran M. Valeurs propres et vecteurs propres en classification hiérarchique. *RAIRO Informatique Théorique* 1976, 10:39–46.
50. Xu R, Wunsch DC. *Clustering*. IEEE Computer Society Press; 2008.

51. Grabusts P, Borisov A. Using grid-clustering methods in data classification. In: *PARELEC '02: Proceedings of the International Conference on Parallel Computing in Electrical Engineering*. Washington, D.C.: IEEE Computer Society; 2002.
52. Wang W, Yang J, Muntz R. STING: a statistical information grid approach to spatial data mining. In: *VLDB '97: Proceedings of the 23rd International Conference on Very Large Data Bases*. San Francisco, CA: Morgan Kaufmann Publishers Inc.; 1997, 18–195.
53. Hinneburg A, Keim D. Optimal grid-clustering: towards breaking the curse of dimensionality in high-dimensional clustering. In: *VLDB '99: Proceedings of the 25th International Conference on Very Large Data Bases*. San Francisco, CA: Morgan Kaufmann Publishers Inc.; 1999, 506–517.
54. Gan G, Ma C, Wu J. *Data Clustering Theory, Algorithms, and Applications*. Philadelphia, PA: SIAM; 2007.
55. Schikuta E. Grid-clustering: an efficient hierarchical clustering method for very large data sets. In: *ICPR '96: Proceedings of the 13th International Conference on Pattern Recognition*. Washington, D.C.: IEEE Computer Society; 1996, 101–105.
56. Sheikholeslami G, Chatterjee S, Zhang A. Wavecluster: a wavelet based clustering approach for spatial data in very large databases. *VLDB J* 2000, 8:289–304.
57. Chang J-W, Jin D-S. A new cell-based clustering method for large, high-dimensional data in data mining applications. In: *SAC '02: Proceedings of the 2002 ACM Symposium on Applied Computing*. New York: ACM, 2002, 503–507.
58. Park NH, Lee WS. Statistical grid-based clustering over data streams. *SIGMOD Rec* 2004, 33:32–37.
59. Ester M, Kriegel H-P, Sander J, Xu X. A density-based algorithm for discovering clusters in large spatial databases with noise, In: *Second International Conference on Knowledge Discovery and Data Mining*. Menlo Park, CA: AAAI Press; 1996, 226–231.
60. Sander J, Ester M, Kriegel H-P, Xu X. Density-based clustering in spatial databases: the algorithm GDBSCAN and its applications. *Data Min Knowl Discov* 1998, 2:169–194.
61. Xu X, Jäger J, Kriegel, H-P. A fast parallel clustering algorithm for large spatial databases. *Data Min Knowl Discov* 1999, 3:263–290.
62. Zaïane OR, Lee C-H. Clustering spatial data in the presence of obstacles: a density-based approach. In: *IDEAS '02: Proceedings of the 2002 International Symposium on Database Engineering and Applications*. Washington, D.C.: IEEE Computer Society; 2002, 214–223.
63. Dash M, Liu H, Xu X. $1 + 1 > 2$: Merging distance and density based clustering. In: *DASFAA '01: Proceedings of the 7th International Conference on Database Systems for Advanced Applications*. Washington, D.C.: IEEE Computer Society; 2001, 32–39.
64. Xu X, Ester M, Kriegel H-P, Sander J. A distribution-based clustering algorithm for mining in large spatial databases. In: *ICDE '98: Proceedings of the Fourteenth International Conference on Data Engineering*. Washington, D.C.: IEEE Computer Society, 1998, 324–331.
65. Hinneburg A, Keim DA. A density-based algorithm for discovering clusters in large spatial databases with noise. In: *Proceeding of the 4th International Conference on Knowledge Discovery and Data Mining*. New York: AAAI Press; 1998, 58–68.
66. Wang L, Wang Z-O. CUBN: a clustering algorithm based on density and distance. In: *Proceeding of the 2003 International Conference on Machine Learning and Cybernetics*. IEEE Press; 2003, 108–112.
67. Murtagh F, Downs G, Contreras P. Hierarchical clustering of massive, high dimensional data sets by exploiting ultrametric embedding. *SIAM J Sci Comput* 2008, 30:707–730.
68. Contreras P, Murtagh F. Fast hierarchical clustering from the Baire distance. In: Hocarek-Junge H, Weihs C, eds. *Classification as a Tool for Research*. Berlin: Springer; 2010, 235–243.
69. Contreras P. *Search and Retrieval in Massive Data Collections*. PhD Thesis. Egham, UK: Royal Holloway, University of London; 2010.

