Improving Visualization of Large Hierarchical Clustering

Gilles Bisson, Renaud Blanch

Improving Visualization of Large Hierarchical Clustering

Gilles Bisson, Renaud Blanch
CNRS, UJF-Grenoble 1
AMA, IIHM teams - LIG Laboratory - UMR 5217
BP 53 - F-38041 Grenoble Cedex 9, France
{gilles.bisson@imag.fr, renaud.blanch@imag.fr}

***Abstract*— The classical representation of a binary tree generated by a hierarchical clustering is a node-link-based visualization denoted as a dendrogram. It allows users to explore in a simple way the clusters and the relationships between instances. However, exploration of large dendrograms is known to be difficult due to the graphical and cognitive information overload involved. Here, we discuss the current approaches and we introduce Stacked Trees, a new Focus+Context visualization technique that allows the exploration of the hierarchical clustering of up to fifty thousands nodes on a standard-sized screen.**

Keywords-component: Hierarchical Clustering, Large Hierarchies, Stacked Trees, Data Mining, Chemoinformatics.

I. MOTIVATIONS

A. The Visualization Challenge

Clustering is a classical explanatory approach (see among many others: [5], [30]) helping to understand and to explore large databases. Nevertheless, a practical drawback of this approach lies in the fact that the validation process is not as straightforward as for the supervised learning methods where it is possible to evaluate a *generalization error* according to a target concept. Several methods are possible [9], namely:

- To use an internal criterion to measure the relevance of the clusters such as their degree of overlap or their cohesiveness. Still, these measures are quite subjective since they are based on some biased ideas about the kind of results we are waiting for.
- To select a supervised database whose category labels are known, then to cluster this database by ignoring these labels and to measure the correlation between existing and learned categories. While this approach is useful to test or setting up an algorithm, it is not relevant in practice since the clustering step aims at discovering some new "surprising" clusters having some unknown properties.
- To work with an expert of the application domain that will explore, by hand, the clustering structures to identify the most meaningful groups. On the one side, in an explanatory process, this approach is the most adapted, but on the other side it is time consuming and requires considerable human resources. Thus, to be successful, one needs to provide to the expert some tools allowing her/him to explore the clustering in order to identify and to interpret the relevant clusters.

In such context, Agglomerative Hierarchical Clustering (AHC) is a well-suited method to provide to the experts some relevant information for analyzing data. Indeed, the AHC procedure organizes data in an intuitive and interpretable way for human being, namely a binary tree. In such structure all degree of generality are present from the basic instances to the most general cluster. However, visualization of large sized trees is known to be difficult since the number of leaves grows exponentially with the depth of the tree. Practically, when dealing with a dendrogram containing more than a few hundred leaves, any node-link representation of the tree becomes unfeasible. Of course, it is always possible to display at one time only a subpart of the structure and to explore the other parts through a combination techniques such as: filtering, distortion, zooming or panning [22]. However, as emphasized by [16], these approaches involve some design tradeoffs and constraints for the user: for instance, the need of integrating the different subviews in the zooming approaches or the need of understanding the meaning of a distorted view. The drawback is that the user loses her/his overview of the whole dataset and that the exploration of the classification needs a training step for the "non specialist" in order to discover the features of the tool. This can become tedious from a cognitive point of view. Therefore, providing a really *static view* able to present a large amount of data in a non-ambiguous, uncluttered, scalable and aesthetic way is a worthwhile challenge.

To deal with this objective various solutions have been proposed by the Information Visualization community (InfoVis) during the two last decades (among many others: [21], [28], [29]). However, in the context of Machine Learning none of these approaches solves the complex problem of offering a complete and an intuitive display of both the hierarchy and the instances at the same time.

B. Application Domain

Before outlining our visualization and comparing it to the existing ones, we first explain a chemoinformatics application whose analysis led us to the current proposal. The ACCAMBA project aimed at developing Machine Learning tools to analyze chemical libraries and to model *High-throughput Screening* results. The screening process is designed to quickly test, by using robotic devices, the bioactivity of a set of molecules, organized within a *chemical library*, on a biological target (enzyme or cell). Each test highlights some tens or hundreds of *active molecules* (named *hits*) representing generally a very small percentage of the

chemical library (<<1%). However, these analyses are only a preliminary step since the identified molecules rarely fulfills the appropriate properties both in terms of *sensitivity* (effective dosage) and *specificity* (lack of side effects). Thus, to design some new therapeutic molecules (*Drug Discovery*), the chemical library must contain a large set of different molecular structures to cover the *chemical space* relevant to study the biological target [23], [12]. In this context, it is crucial to provide to the chemists some interactive tools enabling them to pinpoint the active molecules within this chemical space and to ease the search for related molecules in order to synthesize more efficient compounds.

In the last decade, many similarity/distance measures have been developed between molecules using either features-based or graphs-based representations, among many others: [1], [24], [31], [32], [33]. These measures can be used by any AHC method with the Ward criterion to classify some compound libraries containing up to tens of thousands of molecules. However, as we saw in the beginning of this section, the real benefit of building a dendrogram depends on the ability of the chemists to analyze its meaning and to address some classical exploratory tasks. For instance:

- To identify the relative position of the hits (active molecules) with respect to the main clusters in order to see if their chemical structures are related or not.
- To look at the position of the *hits* within the already known families of molecules in order to know where these molecules are situated in the chemical space.
- To display the shared chemical/physical properties (mass, logP, etc.) of the hits and more generally to see the properties associated to a given cluster.
- To detect unexpected clusters and to analyze if they bring new information or if they just reveal errors or incompleteness in the description of instances.

Obviously, these tasks are not specific to chemistry: exchanging the terms "hits", "molecules", … by the equivalent objects of another application domain, leads to the same kinds of tasks. In each case, the goal is to understand what are the *environment* and the *properties* of a set of objects (instances/clusters) with respect to the others. In terms of visualization techniques, this means that the user must be able to access simultaneously the *local information* contained in the leaves of the hierarchy, (here, the features of molecules), and also the *global information* characterized by the medium and higher levels of the hierarchy in order to identify the relationships between the clusters. This could be seen as a classical *Focus+Context* [15] problem, but in our task we do not have a single focus but several (the hits) at the same time and all the molecules occurring in the same cluster are *a priori* equally interesting to explore. Thus, the two kinds of information we need to represent are more corresponding to two related contexts with a different level of granularity. However, the notion of *focus* will be reintroduced in Section IV when describing the prototype.

The rest of the paper is organized as follows. In Section II, we will describe the state of the art concerning the visualization techniques of a dendrogram. Then, in Section III we will present a new visualization method named "Stacked Trees" and we will discuss its benefit in terms of *information density* with respect to the other approaches. Finally, in Section IV we will present the prototype we developed and we will describe the interactions between the user and this tool.

II. STATE OF THE ART

A. Visualization in Chemistry

So far, many tools have been developed to visualize a chemical space. A classical approach, both in biology and chemistry, is to use *heat-maps* to display the knowledge associated to a set of objects (biological experiments, genes or molecules) described by a large range of properties ([18], [2]) or more directly, to display the values of a similarity matrix. These approaches use the fact that rows and columns of the heat-map can be sorted so that the map topology remains close (or similar for a dendrogram) to those of the chemical space. In this context, adaptive Self Organizing Maps (SOM) are widely used for drug discovery since they allow simultaneously to categorize and to view the data according to the best topology. However, these approaches do not visualize the hierarchical information, although some recent systems, such as TS-SOM [25], propose a multi-scale approach of the SOMs. Moreover, in the case of these self-adaptive approaches the user does not always have an access to the molecular level since each area of the map corresponds to a cluster of objects. HCE (Hierarchical Clustering Explorer) [34] or more recently [17] which is based on an hyperbolic tree representation propose a more integrated approach, in which each area of the screen represents either a part of the hierarchy or the average values of a set of neighboring objects. But again, in both cases, the representation is too dense and does not fulfill our needs.

In order to visualize large hierarchies, another option, often used with graphs [22], is to reduce their size. These methods are either based on a preprocessing of the dendrogram to filter or to aggregate the less informative clusters/instances with respect to some chemical criteria; or based on a preprocessing of the instances in order to reduce the size of the database. Such approach is used by [8] to detect the Maximum Common Substructures (MCS) (or the *scaffolds* in [35]) between molecules and to perform a hierarchical clustering of these MCS. Of course, some information about the detailed structure of the molecules is lost since the leaves of the tree are corresponding to clusters of molecules (the MCS). This is annoying with the screening data since this implies that the selected MCS are the "right features" to organize the molecules. However, in practice, the right criterion depends both on the biological target and on the assumptions of the chemist analyzing the results.

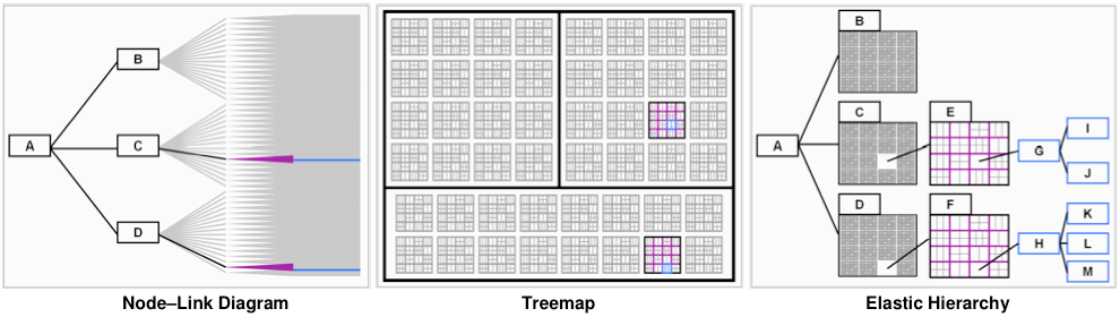

Figure 1. Elastic Hierarchies try to combine node-links and space-filling techniques (extracted from [37])

B. *Generic Visualization Methods*

As said in the introduction, hierarchies are general and intuitive structures allowing us to represent a wide range of phenomena. Thus, countless studies have been conducted in the domain of InfoVis to visualize large hierarchies and to help to compare them like in *TreeJuxtaposer* [26]. The recent survey of [22] provides a wide overview of the representation paradigms, and explores them according to different analysis criteria such as the complexity, the user interaction, etc. One can divide most of the existing representation techniques into three categories: *node-links*, *space filling* and *hybrid*. In the first case, the nodes and their relationships are explicitly drawn on the screen. In the second case, the relationships between clusters are expressed through a set of nested shapes using the full area of the screen. Finally, the hybrid methods try to combine the strengths of each of the two previous approaches.

Hyperbolic trees [21] can be seen as a specific implementation of the *fisheye framework* [15], [16]. They belong to the *node-links* techniques and consist in projecting the whole hierarchy in a non-Euclidean space. This allows the user to explore locally (*focus*) the data by enlarging some parts of the tree without losing the global picture of the data (*context*). However, even if these approaches can be used to display a larger number of objects than a classical "Euclidian" hierarchy, the filling of the available space remains rather small and limits the maximum number of objects in the best cases up to several thousands as in FSVIZ [10]. *Cone Trees* [28] are based on a 3D representation increasing slightly the number of visible clusters up to about ten thousand, but with some drawbacks like the concealing of many nodes by the cones in the foreground and some visual clutter making the exploration of the structure tough. More recently, *SpaceTrees*, introduced by [27], proposes a dynamic approach to the visualization problem in which the different parts of the hierarchy are dynamically reconfigured while browsing the structure through the folding/unfolding of subtrees. In this way, the user can optimize the screen layout according to her/his needs. However, if all these approaches allow keeping an explicit representation of the relationships between the clusters, we must notice that 1) it is impossible to access the information contained in the leaves without moving within the structure and 2) the screen layout limits again the number of visible items to some thousands.

TreeMaps belong to the *space filling* category and were initially proposed by [29] in order to display the full content of a hard disk. In this layout, the hierarchy levels are represented by a sequence of nested rectangles allowing both an optimal use of the display space and the visualization of the low-level information (instances) in a homogeneous way. Moreover, by using the computational power of modern GPUs, we can manage interactive and zoomable maps containing millions of objects as in [7], [13]. Nevertheless, the TreeMaps have also two drawbacks: on the one hand, for the novice user, the hierarchical structure is quite hard to perceive and it is very difficult to distinguish among the different levels of the dendrogram; and on the other hand, the relative position of the blocks (i.e. the clusters) on the screen is not necessarily very intuitive and requires some practices, even if some studies, like those about the *Voronoi TreeMaps* [3] and the *Jigsaw maps* [36], are interesting from this point of view, since they propose a more natural topology of the clusters. While the TreeMaps offers an unmatchable density of information, they are not well adapted to our problem.

With his *Elastic Hierarchies*, [37] proposes an interesting hybrid approach combining on the same screen a node-link representation and a set of TreeMaps (Fig. 1). In this tool, as with the *SpaceTrees*, the user can adapt dynamically the layout in order to fulfill her/his needs for each part of the hierarchy. In this way, one theoretically preserves the advantages of both approaches: interpretability of the hierarchical structure and compactness of the data. However, the criticism about the arbitrary position of the clusters in the TreeMaps remains, in a lesser extent since the map are smaller and the optimal use of such kind of system requires some training from the user. Another hybrid approach exists in other tools such as *NodeTrix* of [18] which allows representing the content of large social networks by combining the visualization of some noticeable links within the network with the display of the adjacency matrices.

III. THE *STACKED TREES* VISUALIZATION

A. *Main Principles*

The *Stacked Trees* method proposed in this paper also belongs to the hybrid representation family, but it is based on a simpler paradigm than the Elastic Hierarchies in the sense: 1) that the user does not have the full ability to select his/her representation layout at each level of the dendrogram, there are only two parts 2) that the visualization layout remains close to the one of classical dendrogram, thus simplifying the learning curve and 3) that the data organization we use to condense the information is not based on a 2D structure as in the TreeMaps but rather on a simplified *1D structure* that we

call “stacks”. It worth noting that similar ideas were proposed by [18] in order to visualize in a compact way the content of large multi-attributes databases. However, this previous work does involve neither notions of hierarchies nor clustering. As we are going to see, this representation provides both a good interpretability of the clustering results, as well as an interesting optimization of the nodes layout (instance and clusters).

The basic idea of the Stacked Trees is the following: as we discussed in Section I.B, when a chemist wants to analyze a hierarchical clustering she/he needs mainly to access simultaneously 1) to the local information contained in the leaves of the hierarchy (here the features of the molecules); and 2) to highest levels of the hierarchy in order to grasp the overall organization of the clusters. In other terms, the medium part of the hierarchy is not very informative. Thus, as shown in Fig. 2, the idea is to suppress this part and to organize all the leaves (instances) belonging to a given *subtree* in a form of a vertical 1D structure corresponding to a *stack*. Therefore, in our terminology, words *stack* and *subtree* are synonyms.

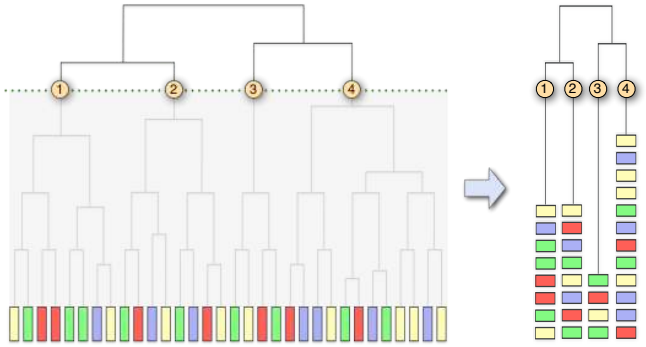

Figure 2. Stacked Trees (right part) keep the highest levels of the dendrogram and the leaves. Intermediate levels are suppressed and leaves are organized into vertical “stacks” increasing the information density.

Beyond the compactness argument, this organization under the form of stacks has also a very interesting property: it allows a dynamic reordering of the instances on the screen in the most relevant way according to the needs of the expert. Indeed, in various application domains, *features* describing the instances are numerical or more generally they belong to an ordered type whose elements can be sorted. This is the case in our chemoinformatics application in which the molecules can be ranked along many dimensions such as: bioactivity intensity, mass, hydrophobicity, etc. Thus, 1D representation is perfectly suited, as we will see in Section IV.C, to organize the information in a comprehensive way for the user. For example, to visualize how the set of molecular masses are distributed within a cluster or to show the similarity between hits and other molecules. The 1D representation is also perfectly fitted to display clustering result since it corresponds to an ordered sequence of leaves (this order can be improved as discussed in Section IV.C). In this way we avoid the problem we evoked about the TreeMaps concerning the arbitrary — or at least difficult to explain— positions of the clusters/instances in the maps.

Finally, we must point out that the Stacked Trees compactness is quite high and intermediate between those of TreeMaps and of node-links-based representations. We evaluate this compactness, expressed in term of *viewable instances*, in the next section.

B. *Information Density Analysis*

The formal evaluation of visualization method usefulness is rather difficult and many metrics has been proposed (see [6] and [22] for recent surveys about these aspects). In this paper, according to our goal of maximizing the quantity of information displayed at the same time, we compare the number of instances and the number of levels that can be simultaneously displayed in the case of a rather “well balanced” hierarchy (Table I). Our baseline is a 24-inch classical screen containing about 2-megapixels (2000px width and 1000px height).

First of all, we need to decide the minimum number of pixels needed to represent the information contained in an instance. To represent in a compact way a tuple <*feature, value*> coding one information of a given instance, we can use a *color code*, that can be either discrete or a gradient. In practice, to be easily visible (and selectable with a standard pointer: mouse, stylus or finger), we make the assumption that each value must fill at least an area of about 3x3 pixels on the screen. Moreover, it is necessary to separate the different instances with 1 pixel in order to stay readable. Consequently, to display each instance we will devote an area of 16 pixels (4x4) on the screen.

As we see in Table I, in the TreeMaps paradigm, the whole screen can be used to display the instances since the hierarchical structure is “implicit”. This allows representing up to 125.000 instances at the same time. For the node-links representation we consider that 1) we use a simple Euclidian projection, 2) that there is not overlapping between links and nodes and 3) that the nodes are aligned along the 4 edges of the screen corresponding to a total width of 5000 pixels. Under these hypotheses, up to 1250 instances (items) can be displayed at the same time.

TABLE I. COMPARISON OF THE INFORMATION DENSITY

Criteria	TreeMaps	Nodes-links	Stacked Trees
Usable area for data (in blue)	Instances area	Instances area Tree structure	Tree structure Instances area
#pixels	Fullscreen: 2Mpx	Edges: ~5kpx	Bottom: ~0.8Mpx
#items	125.000	1250	50.000
#levels	~17	~10	~9

Finally, with the Stacked Trees representation, the computation is little bit more complex. On the one hand, as in our hypothesis each stack has a 4 pixels width (the size of an instance), it is not possible to put more than 500 stacks on a 24” screen of 2000 pixels width. On the other hand, the vertical area dedicated to display the stacks represents about 80% of the screen, as the top of the screen is needed to draw the structure of the higher levels of the hierarchy. Furthermore, even with a well-balanced hierarchy, the stacks will fill, on average, only half of this area since the clusters

cannot be equally sized (see Fig. 3 which is a typical example). By consequence, the display area devoted to the (value of) instances is about 40% of the screen that is corresponding to 400 pixels height (the 24" is 1000 pixels height) and up to 100 instances per stack. So, about 50.000 instances can be represented on the screen.

This number is interesting since it is very close to *the maximal number of instances* that's can be processed by an AHC method and it is about 40 times higher that the number of instances that would be displayed with a classical Nodes-link representation. Let us notice that the number of visible levels is smaller than for the classical tree, since we display a limited number of stacks. When displaying several features (multivariate representation), the number of instances on the screen will decreases in a linear way like for the TreeMaps.

IV. PRESENTATION OF THE PROTOTYPE

A. General Organization

Fig. 3 represents a screenshot of the current prototype of *Stacked Trees* as used in our chemical application. The central area is composed of two parts corresponding to the Stacked Trees representation. We find a classical hierarchy at the top (part number 5 on the figure) with a standard *combination similarity* scale on the right part and a collection of stacks at the bottom (6) each of them corresponding to a particular subtree. The number of subtrees (or stacks) to display is dynamically controlled by the user through a slider (3) allowing an interactive adjustment of the *cut level* of the hierarchy. In this screenshot, this value can be adjusted between 2 and 64 (the current value is set to 31 in Fig. 3) but the upper limit just depends on the width of the screen. The height of each stack is proportional to the number of elements contained in the subtree. The molecule names of the currently selected stack (here, cluster C2135) are displayed in a scroll list (1) by using a classical *Focus+Context* method [16]. Thus, to get the name of a particular molecule in a stack, the user has to select the corresponding rectangle and the molecule will be highlighted in part (1), the selection process being a bidirectional one. The stacks and molecule currently selected are corresponding to the *focus* in the interface, the other stacks and their relational structure being the *context*. The meaning of the other areas on both sides of the interface will be explained in the rest of this section.

The current prototype has been implemented as a Javascript Web application, which can be used through any browser compliant with the W3C web standards. The layout and graphical properties of all the elements of the interface are controlled through CSS; thus, it is very straightforward to modify the overall aspect. The communication between the interface and the data files containing the information is managed by PHP. Four kinds of information must be provided to the interface:

- The hierarchical structure produced by a clustering tool (the format is similar to the one used by the hierarchical clustering module of R: http://www.r-project.org/). Of course, we can also make use of any kind of hierarchy that would be not the result of a clustering process (ontology, spatial information, …)
- An optional similarity matrix between the instances. This similarity matrix is only used by the Stacked Tree to implement the "nearest neighbors" navigation (Section IV.B) and to know the distance between the prototype of a stack (i.e. the *barycenter* of a cluster) and the other instances. In practice, when we want to represent on a screen a huge hierarchy where the size of the similarity matrix can reaches tens of gigabytes, one can avoid it.
- A database containing the <feature, value> tuples coding the information of the instances (here the physical/chemical features).
- Some domain dependent information. Here, the description of the 2D chemical structures used to display the molecules in parts (7) and (8).

We must emphasize that only the last item is dependent of our chemical application. Indeed, if we change the content of the <feature, value> database, the interface will display other information about the instances. Therefore, the current implementation of the Stacked Trees is totally generic.

B. Visualization of the Features

In the screening problem the notion of *hits* is defined by reference to a (numerical) bioactivity threshold whose value depends both on the biological experiment performed and on the type of information sought during the analysis. Thus, the chemist must be able to tune easily this threshold and to have a quick visual feedback about the number and the position of the hits with respect to other molecules. More generally, there are many other interesting physical and chemical properties that are attached to each molecule, such as: the mass, degree of hydrophobicity (LogP), electronic charge, and so on. Thus, as for the bioactivity threshold it is important to provide an access to this knowledge. In our prototype, the display of these pieces of information is controlled by the content of the database described in the last Section IV.A. Furthermore, the user can customize the way properties are displayed in a homogeneous manner.

TABLE II. DISPLAY RULES OPTIONS

Position	Behavior on part 1	Behavior on part 6
Left	Color of the left part	Color of the left part
Fill	Color of the line	Color of the central part
Right	Color of the right part	Color of the right part
Bold	Name of the instance written in bold	*none*
Italic	Name of the instance written in italic	*none*

Indeed, in the part numbered 2 of the interface (Fig. 3), the user can declare any number of disjunctive *display rules*, classically expressed in the form of triplets *<feature, selector, value>*. For example, if the user define two rules "Bioactivity > 5200" and "Family = Indole" then she/he can associate to each of these rules a position and a color (the possibilities are described in Table II) that changes the way the instances are displayed in parts 1 and 6 of the interface. For instance, in Fig. 4, we colored the molecules whose *typicality* (this concept will be defined later) is smaller than

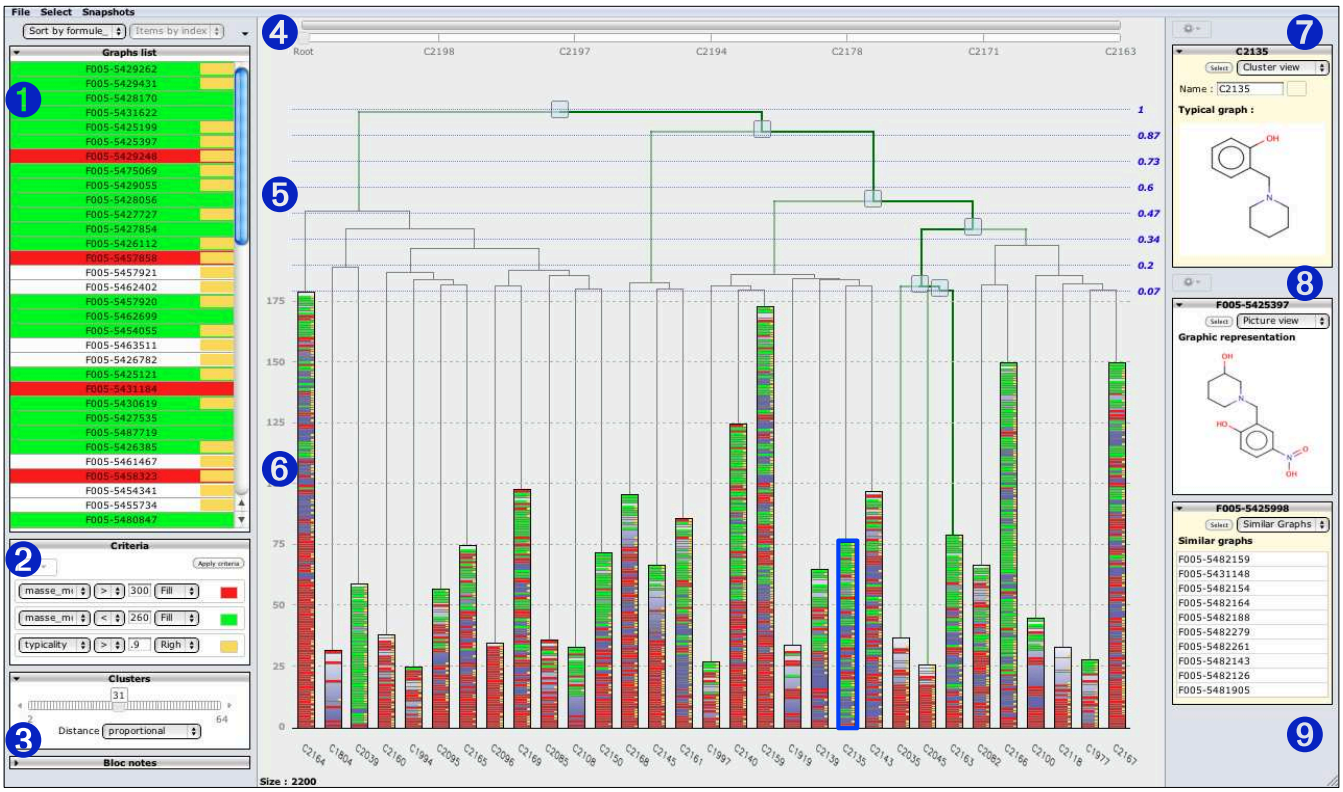

Figure 3. Here is a screenshot of the prototype. The blue numbers highlight the main parts of this interface whose roles are detailed in the paper. The central part contains a Stacked Tree (4,5,6). The left (1,2,3) and right (7,8,9) widgets allow to explore and to control this structure but are independent of the Stacked Trees. While this screenshot has been done on a small 13" screen, 2200 molecule are displayed with for each one up to three properties.

0.7 (orange = right) and whose mass is smaller than 150 the (fill = green) or greater than 250 (fill = red). When the number of instances to display becomes too large, several one can be displayed on the same line since the screen resolution is limited: with a stack of 600px height we can only draw 200 instances (with 3 pixels per instance). In that case, information to display in the stack is determined by the order of the rules: the criterion used is the first one to occur.

Thanks to this mechanism, it is possible to represent a wealth of information in a very compact way and to provide a visual feedback to queries such as: "*Where are situated the hits in the clusters?*", "*Where are located these hits with respect to the molecule having a given mass and hydrophobicity?*", "*What is the homogeneity of the chemical space in terms of masses*", etc. For instance, in Fig. 3, we can easily see that the third cluster (C2039) contains only molecules whose mass is less than 260 (green values) but that this is the opposite for the eighth cluster (C2096).

Finally, by using different colors schema, we can also display the hits corresponding to different values of the bioactivity thresholds. Finally, in chemistry it is crucial to provide to the experts a quick access to the 2D structure of the molecules. In the right part of the interface the user can visualize (7) the most typical molecule of the currently selected stack. She/he can also create as many viewing editors (8, 9) she/he needs to compare different molecules. The drawing of the molecules is performed externally "on the fly" by the application server using the free tool MARVIN by Chemaxon. Some other (not graphical) possibilities are also offered in these parts, such as the ability to view a list of the "nearest neighbors" of the selected molecule (9). This list is also computed on the fly by using the distance matrix used to build up the hierarchical clustering. The user can directly make use of this list to navigate within the hierarchy in an *associative way*. More details about the navigation features will be provided in Section IV.D. Again, it is important to emphasize that while we implemented some parts of this interface to deal with a chemistry problem, the approach is generic. As we saw, the attributes appearing in the display rules (2) are defined in an external database and the viewers used of the parts (7) and (8) can be easily replaced in the interface to generate other kinds of graphical representations (if such drawing make sense for the targeted application, of course).

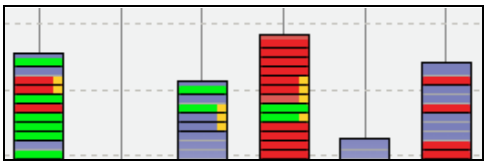

Figure 4. Each element of a stack is a leaf (instance) of the tree that can be currently colored in three parts to express the value of some features.

C. *Instance Ordering in the Stacks*

To help to understand the tree structure and the meaning of the clusters, it is important to allow the user to change his point of view on the data. For instance, in our application, if

she/he decides to display information about the mass of the molecule to see if the "hits" are corresponding to low or heavy molecules, it is interesting to reorganize the content of the stacks by ascending (or descending) molecular mass values. Thus, the order of the instances in the stacks can be modified by the user through the menu at the top of the part (1). There are three possibilities:

- Sorting by the position of the leaves in the hierarchy: in that case, the order is the one computed by the clustering process or more generally as provided in the data set assuming it expresses a similarity scale.
- Sorting by values: when the value of a feature belongs to an ordered type, it is useful to sort the molecules in parts (1) and (6) by increasing (or decreasing) values. The position of the molecule in the stack can be either relative (indexed position) or absolute. In the latter, the top and bottom of the stack are respectively corresponding to the maximal and minimal values of the collection and the stack shows the *real distribution* of the values within the cluster.
- Sorting by typicality: this notion of typically is related to the similarity matrix used during the clustering process. There are two possibilities:

  1. The classification algorithm (AHC) that is included our prototype, automatically determines for each level of the tree the most "typical" instance, by selecting the one minimizing the square distance to the others instances. If the user activates this option, the molecules of the stacks are organized as follows: the bottom of the stack contains the most typical molecule and the other molecules are placed above with an offset proportional to the distance to this typical molecule. In this way, we can easily see if the structure of the hits are typical or not in the clusters.
  2. The user can select any instance in a stack and reorganize all the others with respect to the distance with this one. This is especially useful when looking at the similarity between an interesting instance (e.g. a hit) and the other instances of the stack.

Finally, a well-known drawback of the hierarchical clustering, due to the use of an *ultrametric distance*, is the fact that at the end of the process, the order of the instances is not optimal in the sense that, if we take the sequence of all the leaves as provided by the clustering system $\{L_1, L_2 \ldots L_n\}$ and that we compute the sum of the similarities between the successive neighbors L_i and L_{i+1}, this sum is not maximal. Thus, to maximize the relevance of the hierarchical structure, it is interesting to reorder the whole set of instances at the end of the clustering step by using a serialization algorithm [4], [14] seeking for an order on the leaves maximizing the sum of the similarities between consecutive neighbors. This operation is in $O(n^3)$.

D. *Navigating in the Hierarchy*

Obviously, even if the main objective of Stacked Trees is to focus on the lower and upper part of a hierarchy and to provide a *static view* able to display a large amount of data, it is nevertheless useful to allow the user to explore a particular

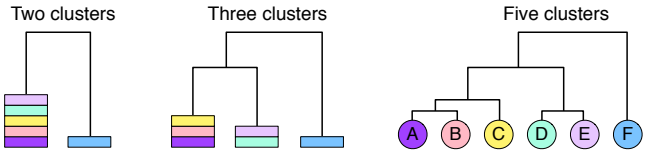

Figure 5. Transformation steps to turn a Stacked Tree into a classical hierarchy (stacks with just one instances).

part of the clustering, to increase the readability of the information. Moreover, as we saw in Section III.B, Stacked Trees are not able to display at the same time several hundred of thousands of instances. Thus, the user can, at any time, select a new root for the hierarchy by simply clicking on one node (box) in the upper part (5) of the interface. It is possible to recursively go down into the tree and then to explore the subclasses. Of course, when going down into the hierarchy, the stacks contain fewer and fewer molecules and at the end, when the number of visible molecules becomes equal to the number of classes selected, the behavior of the interface is similar to the one of a standard tree viewer: each stack is corresponding to a single leaf (Fig. 5). In this sense, Stacked Trees approach can be seen as a natural generalization of the standard node-link representation.

Finally, when the user changes the root of the display, it is important to keep track of the location of the selected subtree with respect to the whole hierarchy (Fig. 6). That is the goal of the two horizontal bars (part 4 in Fig. 3) situated at the top of the interface. They indicate, respectively, the region of the tree that is currently visible and the list of nodes that has been recently explored between the root of the hierarchy and the most specific cluster. This path is also emphasized in green in the hierarchy.

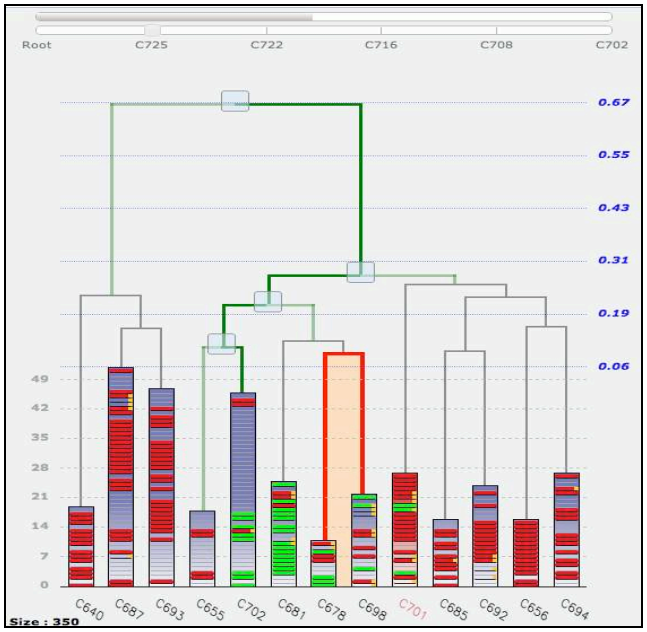

Figure 6. Navigation example. This figure shows the path in the hierarchy between the current root (here the cluster C725) and the most specific stack that has been previously explored (C702) by the user.

Furthermore, the second bar includes a slider allowing the user to dynamically change the position of the current root of tree (the root level is the leftmost position). This is corresponding to a vertical move in the hierarchy. Finally, when increasing or decreasing the number of clusters, the next stack that will be respectively split or merged is shown in red. In Fig. 6, decreasing the number of stacks will lead to combine clusters C678 and C698 that are corresponding to the two most similar clusters at this level.

V. DISCUSSION

In this paper, we present a new hybrid visualization technique, named *Stacked Trees*. It allows to display and to navigate into large hierarchies, composed of up to 50 000 instances and it provides a good alternative to TreeMaps for this task. To the best our know knowledge, this kind of layout was never proposed neither in the Information Visualization nor Machine Learning literatures.

This visualization paradigm has several nice advantages. Firstly, its layout is simple to understand and it can be seen as a natural generalization of the classical (node-link) visualization of the dendrograms thus easing the learning curve for the user. Secondly, the density of information provided is quite high. Thirdly, from the complexity point of view, once the clustering has been done (but it is not really a part of the approach), all the procedures used to draw the nodes on screen are linear in terms of number of instances. Although this work has been initially done to help the analysis of clustering results in chemistry, the approach is generic and modular enough to bring a new solution to many other application domains. Moreover, this visualization method can be used in all kinds of problems involving the exploration of a large hierarchy of knowledge (e.g., ontologies), independently of the fact it has been produced by a clustering tool or not. A video of the current prototype can be seen on the authors' website[1].

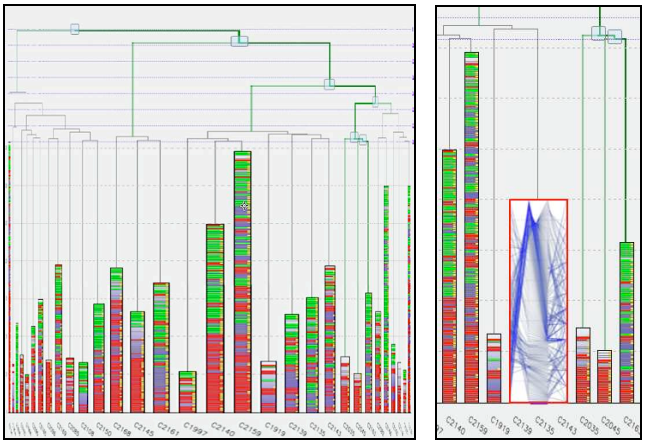

Figure 7. Two mockup examples: a) the Fisheye is centered on the current focus increasing the readability of the data; b) a parallel coordinates view of the current stack allowing to compare the values of the instances.

[1] http://membres-liglab.imag.fr/bisson/research/Stacked-Trees.mov

Nevertheless, several improvements can be proposed to this new approach. Thus, even if we focused in this paper on the importance to provide to the users a *static view*, able to display a large amount of data in an uncluttered, stable and scalable way, it would be interesting to implement most sophisticated tools to navigate in this structure. For instance, the classical scroll list (part 1 of Fig. 5) could be suppressed by implementing an optional *Fisheye view* centered the *focus* point (stack and/or instance). It would allow displaying more information (Fig. 7, left part).

Moreover, stacks are currently used to display the instances of the clusters in a very simple way. By seeing the stacks as a very general "container", many other possibilities could be explored. First, it would be interesting to improve the representation of the features, for example in order to compare the distribution of multivariate data. In such case, it would be possible to embed a Parallel Coordinates view in the stack having the focus (Fig. 7, right part). Second, as stacks are corresponding to subtrees, it would be useful to represent the their internal structure, by using some gradient density indicating the repartition of a given attribute.

Another more fundamental aspect concerns the extension of this stack-based approach toward more complex structures than trees. For instance, a generalization to planar graphs seems very possible. Finally, two new applications domains are under investigation. 1) The exploration of large opening game libraries, to represent if a sequence of moves leads rather to winning or to loosing positions. 2) The exploration of spatio-temporal data in the frame of a GIS.

ACKNOWLEDGMENT

Many thanks to Ludovic Patey who implemented the first release of the Stacked Trees Web interface.

REFERENCES

[1] Aci S., Bisson G., Roy S. and Wieczorek S. 2007. Clustering of Molecules: Influence of the Similarity Measures. In Selected Contributions in Data Analysis and Classification. Springer. p 433-445.

[2] Auman J., Boorman G., Wilson R., Travlos G. and Paules R. 2007. Heat map visualization of high-density clinical chemistry data. Physiological genomics 31(2):352-6.

[3] Balzer M., Deussen O. and Lewerentz C. 2005. Voronoi treemaps for the visualization of software metrics. Proceedings of the 2005 ACM symposium on Software visualization. p 165 – 172.

[4] Bar-Joseph Z., Demaine E., Gifford D., Hamel A., Jaakkola T. and Srebro N. K-ary Clustering with Optimal Leaf Ordering for Gene Expression Data. Bioinformatics, 19(9), pp 1070-8, 2003.

[5] Berkhin P. 2006. Survey of clustering data mining techniques. In Grouping Multidimensional Data. Springer Berlin Heidelberg. p 25-71.

[6] Bertini E. 2011. Quality Metrics in High-Dimensional Data Visualisation: An Overview and Systematization. IEEE Transactions on Visualization and Computer Graphics. Vol 17, No 12, p 2203-2212.

[7] Blanch R. and Lecolinet E. 2007. Browsing Zoomable Treemaps: Structure-Aware Multi-Scale Navigation Techniques 2007. In IEEE Transactions on Visualization and Computer Graphics 13(6), Proceedings of IEEE InfoVis 2007, pages 1248-1253.

[8] Böcker A. 2008. Toward an Improved Clustering of Large Data Sets Using Maximum Common Substructures and Topological Fingerprints. J. Chem. Inf. Model. Vol 48 (11), p 2097-2107.

[9] Candillier L., Tellier I., Torre F. and Bousquet O. 2006. Cascade Evaluation of Clustering Algorithms. In proceedings of the 17th European Conference on Machine Learning ECML'2006, Berlin, Germany, 18-22 september 2006, LNAI 4212, p 574-581.

[10] Carriere J. and Kazman R. 1995. Interacting with Huge Hierarchies: Beyond Cone Trees. Proc. IEEE Information Visualization. p 74-81.

[11] Diday E., Lemaire, J., Pouget J. and Testu F. 1982. Eléments d'analyse des données. Edition Dunod-Bordas.

[12] Dubois J., Bourg S., Vrain C. and Morin-Allory L. 2008. Collections of Compounds - How to Deal with them? Current Computer-Aided Drug Design. Vol 4(3), p 156–168.

[13] Fekete J-D. and Plaisant C. 2002. Interactive Information Visualization of a Million Items. In Proceedings of the IEEE Symposium on information Visualization (InfoVis). Washington, DC, 117-126.

[14] Forina M., Lanteri S., Casale M. and Concepción Cerrato Oliveros M. 2007. A new algorithm for seriation and its use in similarity dendrogram. Chemometrics and Intelligent Laboratory Systems. Vol 87. p 262 – 274.

[15] Furnas G. W. 1986. Generalized fisheye views. In Proceedings of the SIGCHI conference on Human factors in computing systems (CHI '86), Marilyn Mantei and Peter Orbeton (Eds.). ACM, New York, 16-23.

[16] Furnas G. W. 2006. A fisheye follow-up: further reflections on focus + context. In Proceedings of the SIGCHI conference on Human Factors in computing systems (CHI '06), Rebecca Grinter, Thomas Rodden, Paul Aoki, Ed Cutrell, Robin Jeffries, and Gary Olson (Eds.). ACM, New York, 999-1008.

[17] Heard J., Kaufmann W. and Guan X. 2009. A Novel Method for Large Tree Visualization. Bioinformatics 25(4):557-558.

[18] Henry N., Fekete J-D and J. McGuffin 2007. NodeTrix: a Hybrid Visualization of Social Networks. IEEE Transactions on Visu & Computer Graphics, vol. 13, nb 6.p1302-1309.

[19] Keim D., Hao M., Dayal U., M.Hsu, J. Ladisch 2001. Pixel Bar Charts: A New Technique for Visualizing Large Multi-Attribute Data Sets without Aggregation. In Proceedings of the IEEE Symposium on Information Visualization 2001 (INFOVIS). IEEE Computer Society Washington DC, USA.

[20] Kibbey C. and Calvet A. 2005. Molecular Property eXplorer: A Novel Approach to Visualizing SAR Using Tree-Maps and Heatmaps. J. Chem. Inf. Model. Vol 45 (2), 523-532.

[21] Lamping J, Rao R. and Pirolli P. 1995. A Focus+Context Technique Based on Hyperbolic Geometry for Visualizing Large Hierarchies. Proceedings of ACM Conference Human Factors in Computing Systems. p 401–408.

[22] Landesberger (von) T., Kuijper, A., Schreck T., Kohlhammer J., van Wijk, JJ. Fekete, J-D and Fellner DW. 2011. Visual Analysis of Large Graphs: State-of-the-Art and Future Research Challenges. Computer Graphics Forum. Volume 30, Issue 6, p 1719-1749.

[23] Lipinski C. and Hopkins A. 2004. Navigating chemical space for biology and medicine. Nature 432. p 855–861.

[24] Mahé P. and Vert JP 2009. Graph kernels based on tree patterns for molecules. Machine learning Journal. Volume 75, Number 1, 3-35, 2009

[25] Matero S., Lahtela-Kakkonen M., Kohonen O., Ketolenen J., Lappalainen R. and Poso A. 2006. Chemical space of orally active compounds. Chemometrics and intelligent laboratory systems. Vol. 84, No 1-2, p 134-141.

[26] Munzner, T., Guimbretière, F., Tasiran, S., Zhang, L. and Zhou, Y. 2003. TreeJuxtaposer: scalable tree comparison using Focus+Context with guaranteed visibility. In proceedings ACM SIGGRAPH. San Diego. p 453-462.

[27] Plaisant C., Grosjean J. and Bederson B. 2002. SpaceTree: Supporting Exploration in Large Node Link Tree, Design Evolution and Empirical Evaluation. IEEE Symposium on Information Visualization (InfoVis). Washington DC, 57-66.

[28] Roberson G., Mackinlay J., and Card S. 1991. Cone Trees: Animated 3D Visualizations of Hierarchical Information. Proceedings of ACM SIGCHI conference on Human Factors in Computing Systems, p. 189-194.

[29] Shneiderman, B. 1992. Tree visualization with tree-maps: 2-d space-filling approach. ACM Transactions on Graphics, 11(1), p.92-99.

[30] Xu, R., and D. Wunsch. 2005. Survey of clustering algo-rithms. IEEE Transactions on neural networks 16:645– 678.

[31] Rupp M., Proschak E. and Schneider G. 2007. Kernel approach to molecular similarity based on iterative graph similarity. Journal of Che- mical Information and Modeling, 47(6) :2280–2286, 2007.

[32] Ralaivola L., Swamidass SJ., Saigo H., and Baldi P. 2005. Graph kernels for chemical informatics. Neural Networks, special issue on Neural Networks and Kernel Methods for Structured Domains,, 18(8) :1093–1110, 2005.

[33] Haranczyk M. and Holliday J. 2008. Comparison of similarity coefficients for clustering and compound selection. J. Chem. Inf. Model., 48(3) :498–508, 2008.

[34] Seo J., Shneiderman B. 2002. Interactively Exploring Hierarchical Clustering Results. IEEE Computer, Volume 35, Number 7, pp. 80-86, July 2002.

[35] Schuffenhauer A., Ertl P., Roggo S., Wetzel S., Koch MA., and Waldmann H. 2007. The Scaffold Tree - Visualization of the Scaffold Universe by Hierarchical Scaffold Classification. J. Chem. Inf. Model. 2007, 47-58.

[36] Wattenberg M. 2005. A note on space-filling visualizations and space-filling curves. In Proc. IEEE Symp. Information Visualization (InfoVis).P 181-186.

[37] Zhao S., McGuffin, M.J., Chignell, M.H. 2005. Elastic hierarchies: combining treemaps and node-link diagrams. in proceedings of Information Visualization. INFOVIS 2005 IEEE Symposium. 57 – 64.