Computer-aided Diagnosis via Hierarchical Density Based Clustering

Tom Obry, Louise Travé-Massuyès, Audine Subias

Computer-aided Diagnosis via Hierarchical Density Based Clustering

Tom Obry[1,2] and **Louise Travé-Massuyès** [1] and **Audine Subias** [1]
[1]LAAS-CNRS, Université de Toulouse, CNRS, INSA, Toulouse, France
[2]ACTIA, 5 Rue Jorge Semprun, 31432 Toulouse

Abstract

When applying non-supervised clustering, the concepts discovered by the clustering algorithm hardly match business concepts. Hierarchical clustering then proves to be a useful tool to exhibit sets of clusters according to a hierarchy. Data can be analyzed in layers and the user has a full spectrum of clusterings to which he can give meaning. This paper presents a new hierarchical density-based algorithm that advantageously works from compacted data. The algorithm is applied to the monitoring of a process benchmark, illustrating its value in identifying different types of situations, from normal to highly critical.

1 Introduction

In data-based diagnosis applications, it is often the case that huge amounts of data are available but the data is not labelled with the corresponding operating mode, normal or faulty. Clustering algorithms, known as non-supervised classification methods, can then be used to form clusters that supposedly gather data corresponding to the same operating mode.

Clustering is a Machine Learning technique used to group data points according to some similarity criterion. Given a set of data points, a clustering algorithm is used to classify each data point into a specific group. Data points that are in the same group have similar features, while data points in different groups have highly dissimilar features. Among well-known clustering algorithms, we can mention K-Means [1], PAM [2], K-Modes [3], DBSCAN [4].

Numerous validity indexes have been proposed to evaluate clusterings [5]. These are generally based on two fundamental concepts :

- compactness, the members of each cluster should be as close to each other as possible. A common measure of compactness is the variance, which should be minimized.
- separation, the clusters themselves should be widely spaced.

Nevertheless, one must admit that the concepts discovered by even the most scored clusterings hardly match business concepts [6] [7]. One of the reasons is that data bases are often incomplete in the sense that they do not include the data about all the influencial attributes. In particular, business concepts are highly sensitive to environmental parameters that fall outside the scope of the considered business domain and that are not recorded, for instance stock exchange. In addition, the clusters corresponding to business concepts may be quite "close" in the data space and the only way to capture them would be to guess the right number of clusters to initialize correctly the clustering algorithm. This is obviously quite hard. Hierarchical clustering then proves to be a useful tool because it exhibits sets of clusters according to a hierarchy and it modulates the number of clusters. Data can then be analyzed in layers, with a different number of clusters at each level, and the user has a full spectrum of clusterings to which he can give meaning.
Hierarchical clustering identifies the clusters present in a dataset according to a hierarchy [8][9][10]. There are two strategies to form clusters, the agglomerative ("bottom up") strategy where each observation starts in its own cluster and pairs of clusters are merged as one moves up the hierarchy. The divise method ("top down") where all observations start in one cluster and splits are performed recursively as one moves down the hierarchy. The results of hierarchical clustering are usually presented in a dendrogram. A dendrogram is a tree diagram frequently used to illustrate the arrangement of the clusters. In order to decide which clusters should be combined or where a cluster should be split, a measure of dissimilarity between sets of observations is required. In most methods of hierarchical clustering, splits or merges of clusters are achieved by use of an appropriate metric like euclidean, manhattan or maximum distance.

Few algorithms propose a density-based hierarchical clustering approach like α-unchaining single linkage [11] or HDBSCAN [12]. In this paper, we present a new hierarchical clustering algorithm, named HDyclee, based on density that advantageously works from compacted data in the form hypercubes. This contribution is an extension of the clustering algorithm DyClee [13], [14], [15]. The purpose of this work is to generate a flat partition of clusters with a hypercube's density level higher or equal to a threshold and to be able to visualize all existant clusters in the dataset with a dendogram by varying the density of the hypercubes present in a group. The value of the algorithm in a diagnosis context is illustrated with the monitoring of a Continuous Stirred Tank Heater benchmark, for which it allows the user to identify different types of situations, from normal to highly critical.

This paper is organized as follows. In section 2 the DyClee algorithm is presented. In section 3 the concepts and

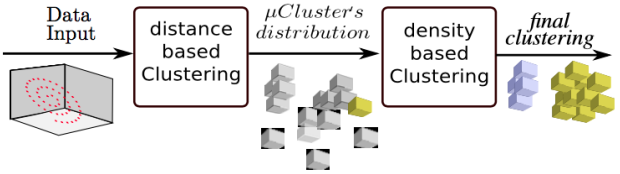

Figure 1: Global description DyClee.

principles underlying Dyclee, like the definition of micro clusters μC, dynamic clusters and the KD-Tree structure, are explained. In the section 4, the hierarchical clustering based-density algorithm is presented. Tests and results are detailed in section 5. The conclusion and perspective for future work end this paper in section 6.[1]

2 Dyclee: a dynamic clustering algorithm

DyClee is a dynamic clustering algorithm which is able to deal with large amounts of data arriving at fast rates by adopting a two stages strategy similar to [16], [17], [18]. The first stage is a fast scale distance-based algorithm that collects, pre-processes and compresses data samples to form so-called *micro-clusters* (μ-clusters). It operates at the rate of the data stream and creates μ-clusters putting together data samples that are close, in the sense of a given distance, to each other. μ-clusters are stored in the form of summarized representations including statistical and temporal information.

The second stage is a slower scale density-based algorithm that groups the μ-clusters into actual clusters that can be interpreted semantically as classes. It takes place once each t_{slow} seconds and analyses the distribution of μ-clusters. The density of a μ-cluster is considered as low, medium or high and is used to create the final clusters by a density based approach, i.e. dense μ-clusters that are close enough (connected) are said to belong to the same cluster. Similarly to [19], a cluster is defined as the group of connected μ-clusters where every inside μ-cluster presents high density and every outside μ-cluster exhibits either medium or low density. The above dense μ-cluster structure allows the algorithm to create clusters of non convex shapes even in high dimensional spaces and it has proved outliers rejection capabilities in evolving environments [18]. In addition, μ-clusters of similar densities can form clusters of any shape and any size.

In DyClee, both stages work on-line, but operate at different time scales. This multi-density feature allows the detection of novelty behavior in its early stages when only a few objects giving evidence of this evolution are present. Figure 1 gives the global description of DyClee.

3 Main principles of DyClee

All the principles explained in this section are from the core algorithm.

3.1 Notion of micro-clusters μC

Considering a d-dimensional object $X = [x^1, ..., x^d]$ marked with a timestamp t_X and qualified by d features, a μ-cluster gathers a group of data samples *close in all dimensions* and whose information is summarized in a characteristic *feature vector* (CF). For a μ-cluster μC_k, CF has the following form:

$$CF_k = (n_k, LS_k, SS_k, t_{lk}, t_{sk}, D_k, Class_k)\,. \qquad (1)$$

where $n_k \in \Re$ is the number of objects in the μ-cluster k, $LS_k \in \Re^d$ is the vector containing the linear sum of each feature over the n_k objects, $SS_k \in \Re^d$ is the square sum of features over the n_k objects, $t_{lk} \in \Re$ is the time when the last object was assigned to that μ-cluster, $t_{sk} \in \Re$ is the time when the μ-cluster was created, D_k is the μ-cluster density and $Class_k$ is the μ-cluster label if known. Using LS_k, SS_k and n_k the variance of the group of objects assigned to the μ-cluster k can be calculated.

The μ-cluster is shaped as a d-dimensional box since the L1-norm is used as distance measure. The distance between an object $X = [x^1, ..., x^d]^T$ and a μ-cluster μC_k, named as $dis(X, \mu C_k)$, is calculated as the sum of the distances between the μC_k vector center $c_k = [c_k^1, \ldots, c_k^d]^T$ and the object value for each feature as shown in equation (2):

$$dis(X, \mu C_k) = L_1(X, c_k) = \sum_{i=1}^{d} \left| x^i - c_k^i \right|. \qquad (2)$$

The data is normalized according to the data context, i.e. the feature range $[min^i, max^i]$ of each feature i, $i = 1, ..., d$. If no context is available in advance, it may be established online. The size of the hyperboxes S^i along each dimension i is set as a fraction of the corresponding feature range. The hyperbox size per feature is hence found according to (3), where ϕ^i is a user constant parameter in the interval $(0, 1)$, establishing the fraction:

$$S^i = \phi^i |max^i - min^i|, \quad \forall i = 1, \ldots, d. \qquad (3)$$

Whenever an object X arrives, the algorithm searches for the closest μ-cluster. Once found, a maximal distance criterion is evaluated to decide whether or not the object fits inside the μ-cluster hyper-box. If the fitting is sufficient the μ-cluster feature vector is updated using the object information; if not, a new μ-cluster is created with the object information using its time-stamp as cluster time of creation.

The density of a μ-cluster μC_k is calculated using the current number of objects n_k and the current hyper-volume of the bounding box $V_k = \prod_{i=1}^{d} S^i$, as shown in (4):

$$D_k = \frac{n_k}{V_k}. \qquad (4)$$

Let μC_{k_α} and μC_{k_β} be two μ-clusters, then μC_{k_α} and μC_{k_β} are said to be *directly connected* if their hyper-boxes overlap in all but φ dimensions, where φ is an integer. The parameter φ, fixed by the user, establishes the feature selectivity.

A μ-cluster μC_{k_1} is said to be connected to μC_{k_n} if there exists a chain of μ-clusters $\{\mu C_{k_1}, \mu C_{k_2}, \cdots, \mu C_{k_n}\}$ such that μC_{k_i} is directly connected to $\mu C_{k_{i+1}}$ for $i = 1, 2, \cdots, n-1$. A set of connected μ-clusters is said to be a *group*.

[1]This work is performed in the framework of a CIFRE Project supported by ACTIA.

3.2 Dynamic clusters

Dyclee is a dynamic clustering algorithm, which means that not only the parameters but the classifier structure changes according to input data in an automatic way. It achieves several cluster operations like creation, elimination, drift, merge, and split. For instance, a cluster is splitted into two or more clusters if, with the arrival of new data, high density regions can be distinguished inside the cluster. In that scenario, dense regions are separated by low density regions, making the cluster no longer homogeneous. Even more, the cluster center could be situated in a low density region, loosing its interpretability as prototype of the elements in the cluster. Splitting the cluster creates smaller homogeneous clusters, completely representative of the belonging samples. An illustrative example of this phenomenon is shown in Figure 2.

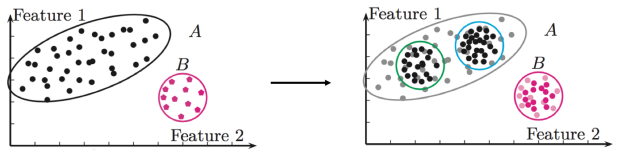

Figure 2: Splitting of cluster A.

3.3 Finding groups of μ-clusters: the KD-Tree structure

To find groups of connected μ-clusters, a KD-Tree [20] is used. A KD-Tree is a binary tree, each of whose nodes represents an axis-aligned hyperrectangle. Each node specifies an axis and splits the set of points based on whether their coordinate along that axis is greater than or less than a particular value. The tree is queried to return only neighbors who are at a maximum distance from a point. A μC_j is the neighbor of the μC_k if the condition in equation (5) is respected :

$$L_\infty = \max_{i=1}^{d} |x_k^i - c_j^i| < r. \quad (5)$$

where d is the number of dimension, c_j^i the center of the μC_j at the dimension i and r the maximal distance from the μC_k. In this context, r is set to ϕ^i.

4 A new hierarchical clustering density-based algorithm

In this section, a new hierarchical clustering density-based algorithm is presented. Inputs are the connections between μ-clusters from the KD-Tree and the output is a flat partition of clusters where all μ-clusters that are in clusters have a minimum density level guaranteed.

4.1 Representation of the μ-clusters connections

The representation of the connections between all μ-clusters can be visualized by a weighted Graph. A weighted Graph $\mathcal{G} = (\mathcal{N}, \mathcal{E}, \mathcal{W})$ is a triplet where $\mathcal{N}$ is a set of nodes. A node n_i corresponds to the μ-cluster μC_i. $\mathcal{E}$ is the set of edges with e_{ij} the edge between the node n_i and n_j, which are unordered pairs of elements of $\mathcal{G}$. Finally, $\mathcal{W}$ is a set of weights on $\mathcal{E}$ with w_{ij} defined in the equation 6.

$$w_{i,j} = \min(D_i, D_j). \quad (6)$$

where the density of a μ-cluster D_i is defined in the equation 4.
An edge e_{ij} between μ-clusters μC_i and μC_j means those are *directly connected* in the sense defined in the section 3.1. If two μ-clusters are not directly connected, there is no edge between them which leads to a Graph that is not full.
The Graph of μ-cluster's connections is built according to the algorithm 1. Neighbors are searched for each μ-cluster with respect to the equation 5 (lines 3 to 6). The function Search_neighbors() is detailed in the algorithm 2. For each μC_k, the distance L_∞ defined in equation 5 is applied where $r = \phi^i$, x_k^i the value of μC_k at the i^{th} dimension and c_j^i the center of the μC_j at the dimension i. The variable Neighbors_of_k contains all the neighbors of μC_k. An edge e_{kj} is added to the Graph $\mathcal{G}$ for each neighbor μC_j of the μ-cluster studied μC_k and the weight w_{kj} is calculated with the equation defined above (lines 7 to 11).

Algorithm 1 Build the Graph of μ-cluster's connections

```
Require: KD-Tree
1:  G = Graph()
2:  Connection = ()
3:  for k = 1 to Number of μ-clusters do
4:      N = []
5:      N = Search_neighbors(k)
6:      Connection[k] = N
7:      for j = 1 to Nbre of neighbors of k do
8:          Weight = min(D_k, D_j)
9:          G.add_edges(k, j, Weight)
10:     end for
11: end for
```

Algorithm 2 Research of a μ-cluster's neighbors

```
Require: KD-Tree, μ-cluster μC_k
1: for j = 1 to Number of μ-clusters do
2:     Neighbors_of_k = []
3:     if max_{i=1}^{d} |x_k^i − c_j^i| < r then
4:         Neighbors_of_k.add(j)
5:     end if
6: end for
7: return Neighbors_of_k
```

Let us consider five μ-clusters μC_1, μC_2, μC_3, μC_4 and μC_5 with their densities $D_1 = 14$, $D_2 = 12$, $D_3 = 3$, $D_4 = 9$, $D_5 = 13$. Figure 3 shows those five μ-clusters.

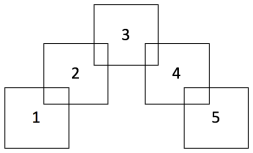

Figure 3: Example of five μ-clusters.

The search of neighbors begins with the densest μ-cluster. In this example, the research starts with the μC_1 as shown in Figure 4. All μ-clusters that satisfy the equation 5 are neighbors of μC_1.

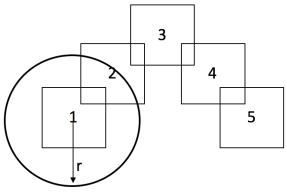

Figure 4: Search of neighbors for the μC_1.

Once neighbors of μC_1 are found, the neighbors of the other μ-clusters are searched. The result is shown in table 1:

μ-clusters	Neighbors
μC_1	μC_2
μC_2	μC_1, μC_3
μC_3	μC_2, μC_4
μC_4	μC_3, μC_5
μC_5	μC_4

Table 1: Neighbors for each μ-clusters

The weights are calculated for every edge following the equation 6 :

$$\begin{cases} w_{12} = \min(D_1, D_2) = \min(14, 12) = 12, \\ w_{23} = \min(D_2, D_3) = \min(12, 3) = 3, \\ w_{34} = \min(D_3, D_4) = \min(3, 9) = 3 \\ w_{45} = \min(D_4, D_5) = \min(9, 13) = 9. \end{cases} \quad (7)$$

The Figure 5 shows the weighted Graph which represents the connections between all μ-clusters in the dataset.

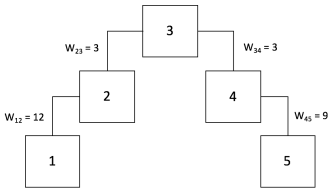

Figure 5: Example 1: Graph of connections between μ-clusters.

4.2 Representation of the hierarchy of clusters

The objective of the algorithm proposed in this paper is to represent and visualize all the possible clusters at different levels of density in a dataset. In contrast to most known hierarchical clustering algorithm, links that relies level l_n and $l_n + 1$ in our new algorithm's dendogram are not based on the distance between objects but on their densities. Furthermore, our proposal is not based on the objects but on μ-clusters that contains the objects to decrease the complexity of calculation. At the root of the tree, there is one cluster composed by all μ-clusters. Each cut in the tree corresponds to a density threshold, i.e each cluster formed below this cut level is composed by μ-clusters that have a density higher to the density threshold. At the bottom of the tree, there is one cluster for each μ-cluster. So root's density level is 0 and the last cut on the tree is $\max(w \in \mathcal{W})$.
Like [12], a variable ε is user parameter evolving in the interval $\varepsilon \in [0, \max(w \in \mathcal{W})]$. For each iteration of ε, all the weights $w \in \mathcal{W}$ are checked. if $w_{ij} < \varepsilon$, the edge e_{ij} is removed. The vertical axis of the dendogram is $1/D$ to keep an ascending direction.

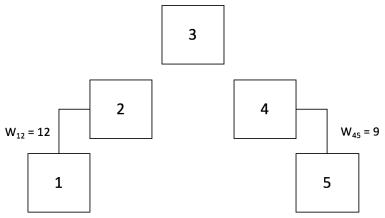

Figure 6: Example 1: Graph of connections between μ-clusters after removed all weight's edges w_{ij} < 3.

Figure 6 represents the case when $\varepsilon = 3$. We can observe two clusters composed by μ-clusters that have their densities strictly higher to three and one μ-cluster alone. This method allows to detect a split of clusters case and to isolate the least denses μ-clusters as early as possible.
Once all values in the interval of ε studied and all edges removed, the dendogram can be generated (see Figure 7).

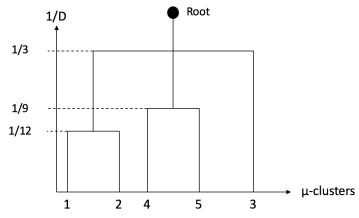

Figure 7: Dendogram of the example 1.

At the density threshold $D = 3$, two clusters composed by μC_1, μC_2 and μC_4, μC_5 are found and the μ-cluster μC_3 is alone. One cluster composed by μC_1 and μC_2 is found and μC_4, μC_5. Then for D=9, one cluster composed by μC_3. Finally, above density $D = 12$, all μ-clusters are alone.

4.3 Extracting a flat partition of clusters

The dendogram generated, a partition of clusters corresponding to a specific density level can be extracted from the Graph of μ-cluster's connections with a density threshold according to the algorithm 2. To find clusters, the edges that have a weight strictly lower than the density threshold are removed (lines 1 to 7). The remaining edges in the Graph have their weights higher or equal to the density threshold

hence μ-clusters forming the clusters are guaranteed to have their densities higher or equal to the threshold. Remaining groups are searched in the Graph. If the size of a group is equal to 1, i.e the group have not connection with other μ-clusters, it is considered as noise. Else, the group is recognized as a cluster (lines 10 to 16).

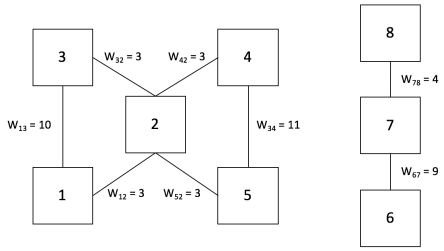

Figure 8: Example 2 : Graph of connections between eight μ-clusters.

An other example is shown in Figure 8 to illustrate this part. Let us consider μ-clusters μC_i, $i = 1, ..., 8$ with their respective densities D_1 = 12, D_2 = 3, D_3 = 10, D_4 = 11, D_5 = 13, D_6 = 9, D_7 = 10 and D_8 = 4.

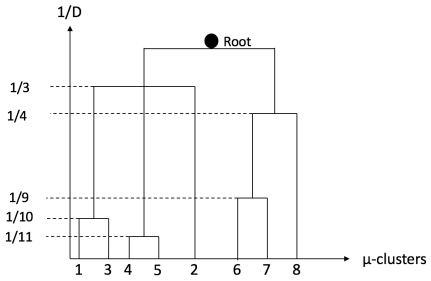

Figure 9: Dendogram of the figure 8.

In this example, the density threshold is fixed to $\varepsilon = 8$. Every group of μ-clusters that is below the density threshold is considered as a cluster. If a μ-cluster does not have connection to the other μ-clusters, then it is considered as noise. The Figure 10 illustrates how to visualize the clusters that have a density strictly above the threshold.

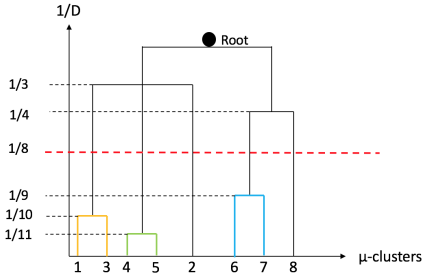

Figure 10: Visualization of clusters in the dendogram that satisfies the $\varepsilon = 8$.

Once the density threshold known, all the edges that have their weights strictly lower are removed to left only groups of μ-cluster to form final clusters. Figure 11 shows the final clusters. The first includes μC_1 and μC_2, the second includes μC_4, μC_5 and the last contains μC_6, μC_7. μC_2 and μC_8 are considered as noise because they do not have any connections with the other μ-clusters.

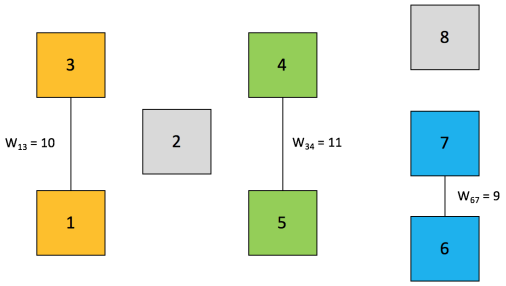

Figure 11: Clusters formed with a density guaranteed strictly above 8.

Algorithm 3 Extract a flat partition of clusters

Require: A density threshold DT, Weighted Graph G

1: **for** ε = 0 to DT **do**
2: **for** all $w(i, j)$ in G **do**
3: **if** $w(i, j) < \varepsilon$ **then**
4: G.removed(e(i,j))
5: **end if**
6: **end for**
7: **end for**
8: Groups = Find_Groups()
9: $k = 0$
10: **for** all g in Groups **do**
11: **if** size(g) == 1 **then**
12: g = Noise
13: **else**
14: $Cluster\_k = g$
15: **end if**
16: **end for**

5 Preliminary tests and results

HDyClee is tested on a benchmark similar to the well known Continuous Stirred Tank Heater (CSTH) of [21]. The CSTH is a stirred tank in which hot and cold water are mixed and further heated using steam. The final mix is drained using a long pipe [14]. Figure 12 shows the structure of the CSTH.

Process inputs are set-points for the cold water, hot water and steam valves. Process outputs are hot and cold water flow, tank level and temperature. Process inputs and outputs represent electronic signals in the range 4-20 mA. The test is done using three output variables: cold water flow CW_{flow}, tank level $Tank_{level}$, and temperature of the water in the tank $Tank_{temperature}$, in the operation mode $OP1$. In this mode, these variables are regulated at the values provided in table 2. The process undergoes several

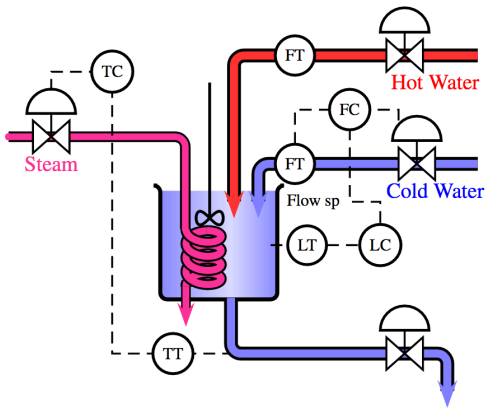

Figure 12: The continuous stirred tank heater.

faults and several repairs that are reported in table 3.

Variable	$OP1$
$CW_{flow}(mA)$	11.89
$Tank_{level}(mA)$	12.00
$Tank_{temperature}(mA)$	10.50

Table 2: Nominal values for the test on CSTH.

The measurements of CW_{flow}, $Tank_{level}$, and $Tank_{temperature}$ are shown in Figure 13 and their recorded values across time constitute the data set for our hierarchical clustering experiment. Sudden changes in the value of regulated variables are indicative of the occurence of some fault or of some fault being fixed. The dataset was generated by simulation.

Event	Description
l_1	Evolving leak starts. Hole diameter goes from 1 to 3, 5mm in 1500 seconds
$\overline{l_1}$	Leak fixed
l_2	A second evolving leak starts. The second hole goes from 0 to 1mm in 1500 seconds
$\overline{l_1 l_2}$	Leaks fixed
s_1	Steam Valve stuck (closed)
$\overline{s_1}$	Valve repaired
s_2	Hot water valve stuck at 10%
$\overline{s_2}$	Valve repaired
l_3	Evolving leak starts. Hole goes from 1 to 2.6mm in 1000 seconds
$\overline{l_1}$	Leak fixed

Table 3: Description of faults on the CSTH system for the operation mode $OP1$.

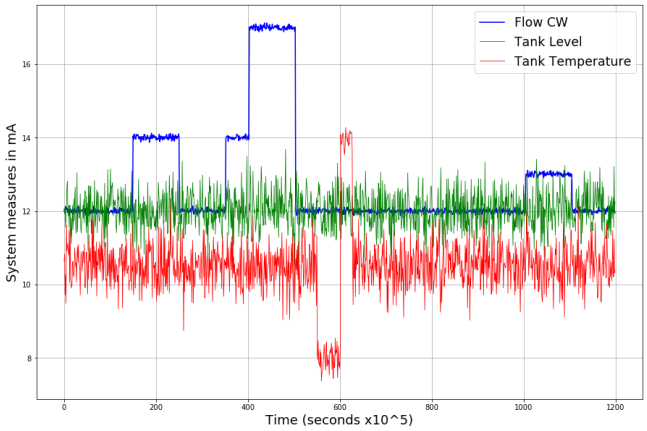

Figure 13: Process measurements for multiple fault scenario.

For this experiment, the radius of research for a μ-cluster's neighbors r is set to 0.06. The parameter φ that defines the number of dimensions that must overlap so that two μ-clusters are considered *directly connected* is set to 0. That means two μ-clusters μC_{k_α} and μC_{k_β} are *directly connected* if their hyperboxes overlap in all dimensions. The parameter ε is set to 0 for results shown in Figure 16 that correspond to the root of the dendogram. For the following graphs, the x-axis is the flow CW_{flow} normalized and the y-axis is the tank temperature $Tank_{temperature}$ normalized. The tank level $Tank_{level}$ is not plotted. Figure 14 shows the graph of connections between μ-clusters when $\varepsilon = 0$. Each red square represents a μ-cluster. Micro-clusters that are not connected to the others are considered as noise. The μ-cluster μC_{293} is connected to μC_{222} and μC_{232}, meaning there are directly connected. μC_{222} is connected to μC_{34} and μC_{232} is connected with μC_{41} and so on. This chain of μ-clusters forms a cluster. Figure 15 shows a generalized dendogram representing the hierarchy of behaviors found in the dataset.

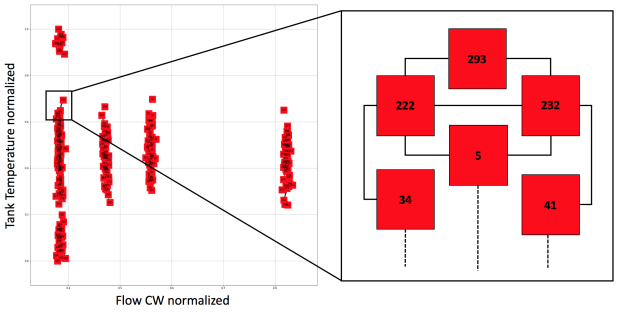

Figure 14: Graph of μ-clusters connections for the study case when $\varepsilon = 0$.

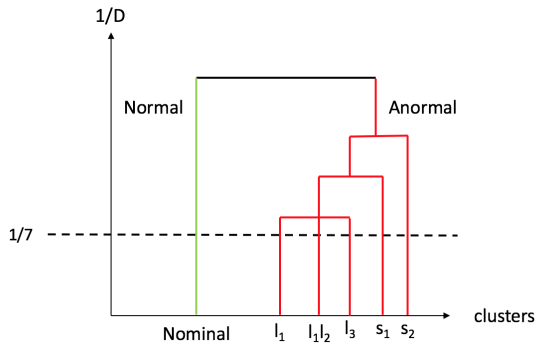

Figure 15: The dendogram with the nominal behavior and the occurrence of faults.

HDyClee detects the 6 main behaviors as shown the Figure 16. The biggest cluster (green cluster) represents the nominal behavior, the blue cluster represents fault s_2, the pink cluster (bottom of the Figure) represents the fault s_1, the grey cluster models the fault l_3, the brown cluster shows the event l_1, and the purple cluster represents l_1 and l_2 present simultaneously. Black points represent μ-clusters that have no connection with other μ-clusters. All the objects inside these μ-clusters are considered as noise.

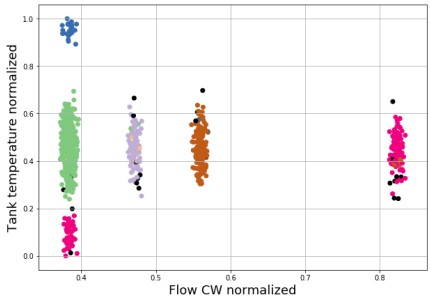

Figure 16: Clusters found by HDyClee algorithm: the nominal behavior (green cluster) and abnormal situations.

It is possible to visualize the most frequent behaviors of the system, in our case the normal behavior and fault l_3. This is reproted in Figure 17. For this purpose, the density threshold is set to ε = 6 by using the dendogram. At this density, the clusters corresponding to other behaviors are considered as noise because their maximal densities are less than 6.

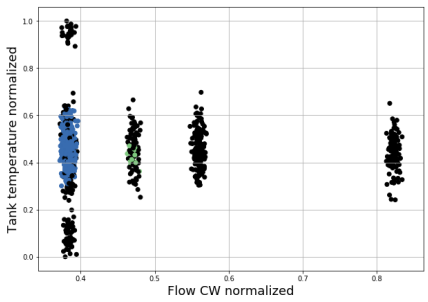

Figure 17: The nominal behavior (blue) and the most frequent fault (green).

To visualize only the nominal behavior, the dendogram must be cut at the density threshold ε = 7 because above this value, the other clusters have no μ-clusters that are connected to each other. This is shown on Figure 19. Figure 18 illustrates the graph of μ-cluster connections after deletion of the edges with a weight $w_{ij} < 7$. Some μ-clusters which were part of the biggest cluster are now considered as noise. Indeed, edges that connected them to other μ-clusters were less than the density threshold.

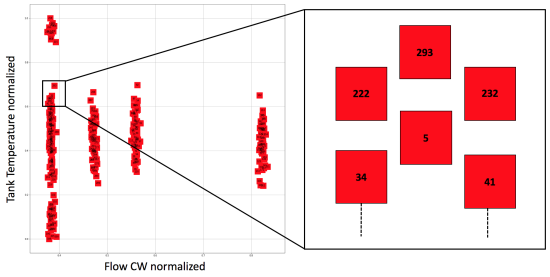

Figure 18: Graph of μ-clusters connections for the study case after removing the weight's edges $w_{ij} < 7$.

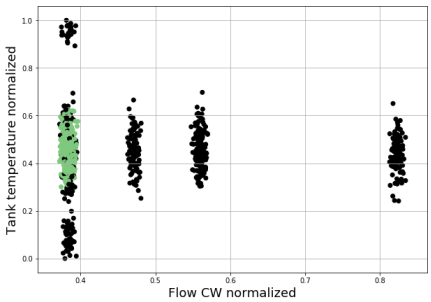

Figure 19: Visualization of the nominal behavior after having cut the dendogram at level ε =7.

6 Conclusion and perspectives

The work presented in this paper proposes a new hierarchical density-based algorithm named HDyclee. The purpose of this algorithm is to extract a hierarchy of clusters that are guaranteed to have a level of density at each layer. Branches in the dendogram do not represent distance between objects but minimum density difference. This approach allows one to identify clusters with the poorest densities and then walk up the hierarchy for higher densities. The algorithm is detailed and tested on a well known monitoring benchmark. HDyClee is able to detect all the behaviors of the process and the user can explore more or less frequent behaviors by cutting the dendogram at different densities.

Next step is to develop experimentations in order to compare this new algorithm with other density-based algorithms. Then the comparative study will include hierarchical and distance-based clustering methods [22], [23]. Several perspectives have been identified for HDyClee, which follow from DyClee properties. In particular, DyClee has a forgetting function that allows to forget μ-clusters which do not receive any data or are not significant (not denses μ-clusters). This function will be included in HDyClee, which

will allow us to produce a dynamic dendogram and then to visualize the evolution of the different behaviors of a system.

References

[1] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.

[2] Leonard Kaufman and Peter Rousseeuw. *Clustering by means of medoids*. North-Holland, 1987.

[3] Zhexue Huang. Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data mining and knowledge discovery*, 2(3):298–304, 1998.

[4] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Knowledge Discovery Data Mining*, volume 96, pages 226–231, 1996.

[5] Maria Halkidi, Yannis Batistakis, and Michalis Vazirgiannis. On clustering validation techniques. *Journal of intelligent information systems*, 17(2-3):107–145, 2001.

[6] Olatz Arbelaitz, Ibai Gurrutxaga, Javier Muguerza, Jesùs M Pérez, and Iñigo Perona. An extensive comparative study of cluster validity indices. *Pattern Recognition*, 46(1):243–256, 2013.

[7] Yanchi Liu, Zhongmou Li, Hui Xiong, Xuedong Gao, and Junjie Wu. Understanding of internal clustering validation measures. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 911–916. IEEE, 2010.

[8] Stephen C Johnson. Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254, 1967.

[9] Robin Sibson. Slink: an optimally efficient algorithm for the single-link cluster method. *The computer journal*, 16(1):30–34, 1973.

[10] Daniel Defays. An efficient algorithm for a complete link method. *The computer journal*, 20(4):364–366, 1977.

[11] Álvaro Martínez-Pérez. A density-sensitive hierarchical clustering method. *arXiv preprint arXiv: 1210.6292*, 2012.

[12] Ricardo JGB Campello, Davoud Moulavi, and Jörg Sander. Density-based clustering based on hierarchical density estimates. In *Pacific-Asia conference on Knowledge Discovery and Data mining*, pages 160–172. Springer, 2013.

[13] Nathalie Barbosa, Louise Travé-Massuyès, and Victor Hugo Grisales. A novel algorithm for dynamic clustering: Properties and performance. In *Machine Learning and Applications (ICMLA), 2016 15th IEEE International Conference on*, pages 565–570. IEEE, 2016.

[14] Nathalie Barbosa. *A data-based approach for dynamic classification of functional scenarios oriented to industrial process plants*. PhD thesis, Université de Toulouse, Université Toulouse III-Paul Sabatier, 2016.

[15] Nathalie Barbosa, Louise Travé-Massuyès, and Victor Hugo Grisales. A data-based dynamic classification technique: A two-stage density approach. In *SAFEPROCESS 2015, Proceedings of the 9th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes*, pages 1224–1231. IFAC, 2015.

[16] Charu C Aggarwal, Jiawei Han, Jianyong Wang, and Philip S Yu. A framework for clustering evolving data streams. In *Proceedings of the 29th international conference on Very large data bases-Volume 29*, pages 81–92. VLDB Endowment, 2003.

[17] Philipp Kranen, Ira Assent, Corinna Baldauf, and Thomas Seidl. The ClusTree: indexing micro-clusters for any stream mining. *Knowledge and information systems*, 29(2):249–272, 2011.

[18] Feng Cao, Martin Ester, Weining Qian, and Aoying Zhou. Density-based clustering over an evolving data stream with noise. In *Proceedings of the 2006 SIAM international conference on data mining*, pages 328–339. SIAM, 2006.

[19] Yixin Chen and Li Tu. Density-based clustering for real-time stream data. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data mining*, pages 133–142, 2007.

[20] Songrit Maneewongvatana and David M Mount. It's okay to be skinny, if your friends are fat. In *Center for Geometric Computing 4th Annual Workshop on Computational Geometry*, volume 2, pages 1–8, 1999.

[21] Nina F Thornhill, Sachin C Patwardhan, and Sirish L Shah. A continuous stirred tank heater simulation model with applications. *Journal of Process Control*, 18(3-4):347–360, 2008.

[22] Adrián Rodríguez Ramos, José Manuel Bernal de Lázaro, Antônio J da Silva Neto, Carlos Cruz Corona, José Luís Verdegay, and Orestes Llanes-Santiago. An approach to fault diagnosis using fuzzy clustering techniques. In *Advances in Fuzzy Logic and Technology 2017*, pages 232–243. Springer, 2017.

[23] Yaguo Lei, Zhengjia He, Yanyang Zi, and Xuefeng Chen. New clustering algorithm-based fault diagnosis using compensation distance evaluation technique. *Mechanical Systems and Signal Processing*, 22(2):419–435, 2008.