CVis - towards a novel visualization tool to explore the relationship between input and output partitions in multi-objective clustering ensembles

Katti Faceli and Tiemi C. Sakata
Department of Computing
Federal University of São Carlos
Sorocaba, São Paulo, Brazil
Email: katti@ufscar.br and tiemi@ufscar.br

Julia Handl
Decision and Cognitive Sciences Research Centre
The University of Manchester
Manchester, UK
Email: Julia.Handl@manchester.ac.uk

***Abstract*—Ensemble methods for clustering take a collection of input partitions, produced for the same data set, and generate an ensemble partition that tries to preserve the information carried in this collective. Acceptance of the resulting partition(s) by decision makers can be a problem, due to the inherent complexity of ensemble techniques, and the associated lack of intuition on how a consensus has been derived from the original set of input partitions. This problem is exacerbated in multi-objective ensemble techniques, which generate a set of non-dominated consensus partitions. In this context, the selection of a final candidate clustering may require additional insight into the relationships between non-dominated output partitions.**

In this manuscript, we describe the first prototype of a novel visualization tool, CVis, which has been developed as a general tool to provide insight into the relationship between any set of partitions of a given data set. We proceed to demonstrate the specific use of this tool in understanding the relationship between the sets of input, the sets of outputs, and the input-output relationships for the multi-objective ensemble technique MOCLE. We discuss how the interlinked analysis of such sets of partitions can shed light onto the functioning, and the strengths and limitations of a particular ensemble technique. In particular, the tool facilitates the visual analysis of the level of support identified for individual consensus clusters, which is helpful in explaining final solutions to a decision maker.

I. Introduction

Ensemble techniques for clustering [1], [2] are a well-established tool to boost the performance of a weak individual clustering algorithm such as k-means, and, more generally, to identify a consensus clustering from any given set of complementary input partitions. In determining the most promising consentient candidate partition, traditional ensemble techniques typically discard original dissimilarity information between data items and, instead, reformulate the clustering problem in terms of the information carried by the collection of input partitions alone.

Ensemble techniques differ in the level at which consensus is identified (which may be e.g. at the level of individual data points or individual clusters), as well as the specific mechanisms used for the identification and / or quantification of consensus, and the construction of consensus partitions. Typically though, mathematically advanced concepts (such as hypergraphs [3]) are employed in the process of ensemble construction. In consequence, the relationship between an ensemble partition and the original input partitions can be difficult to understand, which can limit the extent to which a consensus partition is acceptable to domain experts and interpretable by them.

Visualization tools play a fundamental role in allowing domain experts to understand the implications and basis of a given clustering solution, and have longed played an important role in the analysis of biological data. Examples of prominent visualization approaches used in bioinformatics are the use of dendrograms to highlight hierarchical relationships between entities, and the use of heat maps to emphasize and highlight similarities of entities across a given feature space [4]. These established approaches typically focus on the analysis and insight regarding the structure of a single clustering solution in view of the underlying dissimilarity / feature space. In the context of ensemble techniques, it is an understanding of the relationships between different clustering solutions that becomes of particular importance, but current visualization tools are poorly equipped to provide such insight. Here, we describe the first prototype of a novel visualization tool CVis that is being developed with the aim of addressing this particular gap in the academic literature and of making a practical contribution to the set of tools available for the analysis of clustering solutions and, specifically, ensemble clustering.

A. Multi-objective ensemble clustering

We demonstrate the strengths of CVis in the context of the multi-objective ensemble technique MOCLE [5]. Starting from a set of input (base) partitions, MOCLE uses a multi-objective evolutionary algorithm to optimize a set of two clustering criteria. The variation operator used during the search is a cross-over mechanism that applies the Meta-Clustering Algorithm (developed in [3]) to two candidate partitions at a time. No mutation operator is used.

MOCLE differs from traditional ensemble techniques in two distinct ways:

978-1-4673-8988-4/17/$31.00 ©2017 IEEE

(i) It can be seen as a special type of ensemble technique. Like other ensemble techniques it uses information from the input partitions to constrain the search for possible consensus partitions. However, MOCLE additionally integrates information about the original dissimilarity matrix between data items (rather than to replace this e.g. by the co-association matrix derived from the set of input partitions), and uses this to guide the search and take ultimate decisions on partition quality. This means that cluster boundaries identified but insufficiently represented in the original set of input partitions may still be rewarded and retained.

(ii) MOCLE guides the search process using a pair of two complementary clustering objectives, in order to provide a more comprehensive formulation of the clustering problem than can be achieved using a single clustering criterion [6], [7]. In consequence, MOCLE typically does not identify a single optimal solution but returns a set of ensemble solutions that are mutually non-dominated, for each choice of cluster number.

Both of the above properties contribute to the importance of suitable visualization techniques when analysing the results returned by MOCLE. As a result of (i), the ensemble solutions may vary in the level of support they receive from the original input partitions. As a result of (ii), users of MOCLE are dealing with sets of partitions at both the input and output stage of the algorithm. Additional insight regarding reliability and support can be derived from understanding the relationships between and within these sets, and visualization can contribute to achieving this.

B. Aims and structure of this paper

In this paper, we provide a first description of the principles behind CVis. We use a number of experiments on microarray data to highlight the value of the approach in understanding the relationships within sets of clustering solutions. In particular, we run MOCLE and a more traditional ensemble technique for a variety of input partitions. We proceed to demonstrate how visualization helps highlight the composition of input sets, and helps shed insight onto the impact composition of the inputs has on the output partitions generated by different techniques. Specifically, the remainder of this paper is structured as follows: A brief overview of CVis is provided in Section 2. Section 3 summarizes the experimental setup used in this paper, including a description of the test set employed (gene expression data), the generation of input partitions, the parameterization of ensemble methods, and the index used to evaluate clustering quality. Section 4 provides results for our analysis and Section 5 summarizes our conclusions and highlight opportunities for future development of CVis.

II. Clusters' Visualizer – CVis

CVis (Clusters Visualizer) is an interactive visualization tool designed for the integrated visualization of different types of clustering collections (e.g. strict hierarchies or otherwise related partitions, as well as unrelated partitions).

The key aim of CVis is to facilitate the identification of clusters that re-occur within different partitions, and to ask questions about the frequency and nature of the occurence of such clusters. Concrete questions in the context of ensemble clustering may take the following form: For a cluster present in a given ensemble solution,

- was this cluster present in the original set of input partitions?
- which of the generating methods / algorithms suggested this particular cluster?
- how frequently did this cluster occur in the set of input partitions?
- is this cluster present in multiple of the ensemble solutions?

To answer these types of questions, CVis takes as its input a collection of partitions of a data set. These partitions are visualized in the form of a table with rows representing individual clusters and columns representing blocks of objects. A block of objects is defined as a group of entities that are grouped in the same cluster in all of the partitions that are being visualized (hence, a block may be a singleton, i.e. a single entity). Key to the CVis visualization approach is the feature that the clusters of a given partition are represented independently. In this way, a cluster can be detached from its original partition and shown together with clusters contributed from other partitions, or it may individually be filtered out if it matches a certain criteria. This supports the user in answering the cluster-specific questions that are highly relevant in this setting (see above).

A. Interactivity

Several interaction mechanisms further support the user in exploring the relations among the clusters contained in the collection. Possible interactions that are currently supported include the following:

- To observe how the same group of objects are organized differently in different partitions, by grouping the rows according to the presence of objects in a given block.
- To remove the redundancy present in the collection of partitions by filtering out identical clusters.
- To hide clusters irrelevant to a particular analysis by filtering out clusters with sizes in a given interval.
- To provide an overview of the clusters constituting the individual partitions, by ordering the rows by partition labels (pLabel).
- To explore the relations of the clusters according to their sizes, by ordering them by this feature (cSize).

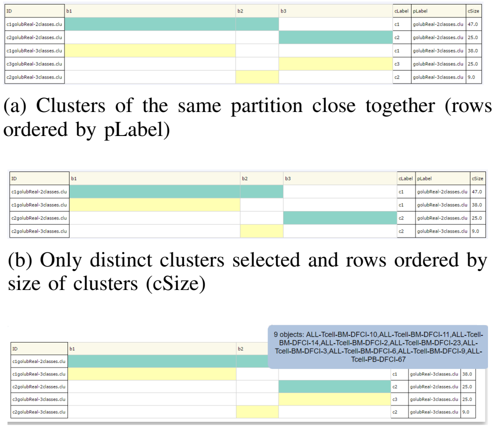
B. Scalability

The size of a column representing a block of objects in the visualization is proportional to the number of objects in the block. The organization of the objects in such blocks helps in summarizing the information which facilitates interpretation but also reduces the sensitivity of the approach to the size of the data set. As a result of this grouping, the suitability of our visualization for large data sets depends primarily on the size of the display device being used and the number of blocks, which is decided by the level of granularity of the input partitions and the level of agreement between them. It does not depend on the size of the data set per se.

Each partition on CVis is represented by a different color, with each cluster shown in a different row. The number of colors available impose a restriction on the number of partitions that can be visualized, but this restriction is comparable to restrictions in many other current tools. Differently from other cluster visualization tools, CVis does not display / colour clusters relative to a reference partition.

C. Current prototype

In order to provide the reader with a visual impression of the design of the current prototype of CVis, Figures 1 (a-c) provide snapshots of the visualization of the two known, valid partitions of the data set Golub (i.e. the two and three cluster solution for this data set). The same data can be explored interactively by the reader at http://lasid.sor.ufscar.br/visualization/cVis/cVis-TP-golub.html.



ID	b1	b2	b3	cLabel	pLabel	cSize
c1golubReal-2classes.clu	🟩	🟩		c1	golubReal-2classes.clu	47.0
c2golubReal-2classes.clu			🟩	c2	golubReal-2classes.clu	25.0
c1golubReal-3classes.clu	🟨			c1	golubReal-3classes.clu	38.0
c3golubReal-3classes.clu			🟨	c3	golubReal-3classes.clu	25.0
c2golubReal-3classes.clu		🟨		c2	golubReal-3classes.clu	9.0

(a) Clusters of the same partition close together (rows ordered by pLabel)

ID	b1	b2	b3	cLabel	pLabel	cSize
c1golubReal-2classes.clu	🟩	🟩		c1	golubReal-2classes.clu	47.0
c1golubReal-3classes.clu	🟨			c1	golubReal-3classes.clu	38.0
c2golubReal-2classes.clu			🟩	c2	golubReal-2classes.clu	25.0
c2golubReal-3classes.clu		🟨		c2	golubReal-3classes.clu	9.0

(b) Only distinct clusters selected and rows ordered by size of clusters (cSize)

9 objects: ALL-Tcell-BM-DFCI-10,ALL-Tcell-BM-DFCI-11,ALL-Tcell-BM-DFCI-14,ALL-Tcell-BM-DFCI-2,ALL-Tcell-BM-DFCI-23,ALL-Tcell-BM-DFCI-3,ALL-Tcell-BM-DFCI-6,ALL-Tcell-BM-DFCI-9,ALL-Tcell-PB-DFCI-67

ID	b1	b2				
c1golubReal-2classes.clu	🟩	🟩				
c1golubReal-3classes.clu	🟨			c1	golubReal-3classes.clu	38.0
c2golubReal-2classes.clu			🟩	c2	golubReal-2classes.clu	25.0
c3golubReal-3classes.clu			🟨	c3	golubReal-3classes.clu	25.0
c2golubReal-3classes.clu		🟨		c2	golubReal-3classes.clu	9.0

(c) Details of the nine objects belonging to block b2

Fig. 1. Snapshots of different orderings / selections in the current CVis prototype.

In this current design of CVis, the blocks of objects are labeled as *bi* in the table header, with *i* corresponding to the number of the block. The column *cLabel* is the cluster label in the corresponding partition, the column *pLabel* reports the labels of the partitions, *ID* provides a unique identifier for each cluster, which is composed of the cluster label and the partition label, and *cSize* gives the cluster size. Complementary information regarding the size of and the objects contained within a given block is shown as a tooltip when the user put the mouse over the corresponding header in the table (Figure 1c).

III. Experimental design

In the experiments included in this paper, we focus on demonstrating the insight a tool like CVis can provide in the context of ensemble clustering – specifically in understanding the relationships between the input partitions and the final ensemble solution(s).

A. Data Set

For the purposes of this study, we employed a gene expression data set that is manageable in size, is widely known and has frequently been employed to illustrate the application of clustering techniques in the context of bioinformatics. Specifically, we make use of the acute leukemia microarray data, which contains data from 72 patients [8], and records gene expression across a set of 3571 genes. Two main classifications of these patients are known and can be considered as known "correct" cluster structures, and clustering algorithms have been successfully supporting these classifications [8]. The first classification corresponds to a two-cluster structure (golubReal-2classes) that differentiates between the acute leukemia AML (Acute Myeloid Leukemia) and ALL (Acute Lymphoblastic Leukemia) types. The structure with three clusters (golubReal-3classes) defines a refinement of the ALL class into the Tcell (T-lineage ALL) and Bcell (B-lineage ALL) lineages.

B. Collections of Partitions

Three different collections of input partitions were used, to illustrate the sensitivity of ensemble techniques to this input and to highlight the role the CVis can play in identifying such sensitivities.

All input partitions were produced using the algorithms k-means (KM) and average-link (AL) [9], available in the software Cluster 3.01 [1]. In both cases, we produced partitions with $k \in [2, 8]$. For KM, we ran the algorithm 30 times for each value of k and selected the partition with the lowest squared error as the partition to be used for the corresponding k. For AL, we generated the hierarchy and cut it in order to produce one partition for each value of k.

Using these partitions, we produced the following three sets of input collections:

- BP-KM-k2-8, which uses the KM partitions only, thus representing a spread of different, non-hierarchical cluster structures;
- BP-AL-k2-8, which uses the output from AL only, thus providing a collection of hierarchical solutions;
- BP-KM-AL-k2-8, which combines the above two sets.

C. Ensemble techniques

Two different ensemble techniques are employed to highlight differences in sensitivity to the composition of the input set. Specifically, we use the multi-objective evolutionary ensemble technique MOCLE, as well as the meta-clustering algorithm MCLA.

MOCLE can be executed online at http://lasid.sor.ufscar.br/mocleproject, and the Matlab/Octave code of MCLA is available at http://strehl.com/soft.html. For both techniques we produced solutions in the interval $k \in [2, 8]$ (equivalent to the range of clusters used for the input partitions). In the case of MCLA, the only parameter required is the number

[1] http://bonsai.ims.u-tokyo.ac.jp/mdehoon/software/cluster/software.htm [10]

of clusters k. For a given choice of k, MCLA will return a partition with at most k clusters. Hence, to produce a collection of partitions, we run MCLA once for each value of k. In the case of MOCLE, we run the algorithm with the parameters summarized in Table I. Here, the Nearest Neighbours parameter determines the number of neighbours considered in the calculation of MOCLE's second clustering objective (Connectivity [7]). This is set to 5% of objects, which translates to four neighbours in the case of the Golub data. All other parameters are self-explanatory.

Parameter	Value
Crossover	MCLA
Crossover Probability	1.0
Minimum k	2
Maximum k	8
Number of Generations	100
Nearest Neighbours (%)	5

TABLE I
MOCLE'S PARAMETERS

Although both ensemble methods are non-deterministic, our analysis here is for single runs of the algorithms only (one for each k in the case of MCLA), as we are interested in understanding the relationships among the partitions they produced instead of evaluating / comparing their performance. Comparisons between different ensemble techniques can be found elsewhere in the literature [11], [3].

D. Evaluation of clustering quality

An established external validation index is used to determine the agreement between a given clustering solution and the two distinct (two and three-class) "true partitions" of the Golub data (TP $k = 2$ and TP $k = 3$). The method selected here is the Adjusted Rand Index (ARI), as this is the method of choice for the comparison of clustering solutions with different numbers of clusters [12]. The Adjusted Rand Index takes as its input a candidate clustering and the correct reference partition. It produces values in the interval $[0, 1]$ with a value of 1 indicating a perfect match between the two partitions.

IV. RESULTS

The overall results obtained in our experiments are summarized in Tables II to IV, which show the best (maximum ARI) solutions contained within both the input collections and the associated output sets of the ensemble methods, broken up by input set. These results emphasize the sensitivity of MCLA to the choice of input set, which arises from its disregard of dissimilarity data. As expected, the use of data-driven objectives in MOCLE can compensate for weaknesses in a particular input set and it therefore shows a more robust performance across inputs. On the other hand, it should be noted that, for MOCLE, the space of possible ensemble partitions is strictly constrained by the cluster boundaries identified in the input set, i.e. no new candidate cluster boundaries can currently be determined by MOCLE's variation operators.

Using these results, we revisit some of the points of interest that were previously highlighted in Section II, and provide concrete examples for the type of analysis that can be conducted using CVis. Note that these illustrative examples have been selected to highlight the specific strengths of our visualization approach, but we do not suggest that the capabilities of CVis are limited exclusively to the type of analysis described here.

Partition Set	TP	k	ARI
BP-KM-k2-8	$k = 2$	2	0.943999
BP-KM-k2-8	$k = 3$	2	0.682689
MOCLE	$k = 2$	2	0.943999
MOCLE	$k = 3$	2	0.682689
MCLA	$k = 2$	3	0.441769
MCLA	$k = 3$	3	0.472003

TABLE II
CHARACTERISTICS OF THE PARTITIONS USING BP-KM-K2-8 AS INPUT

Partition Set	TP	k	ARI
BP-AL-k2-8	$k = 2$	6	0.876025
BP-AL-k2-8	$k = 3$	8	0.798051
MOCLE	$k = 2$	6	0.876025
MOCLE	$k = 3$	8	0.798051
MCLA	$k = 2$	3	0.875978
MCLA	$k = 3$	5	0.816329

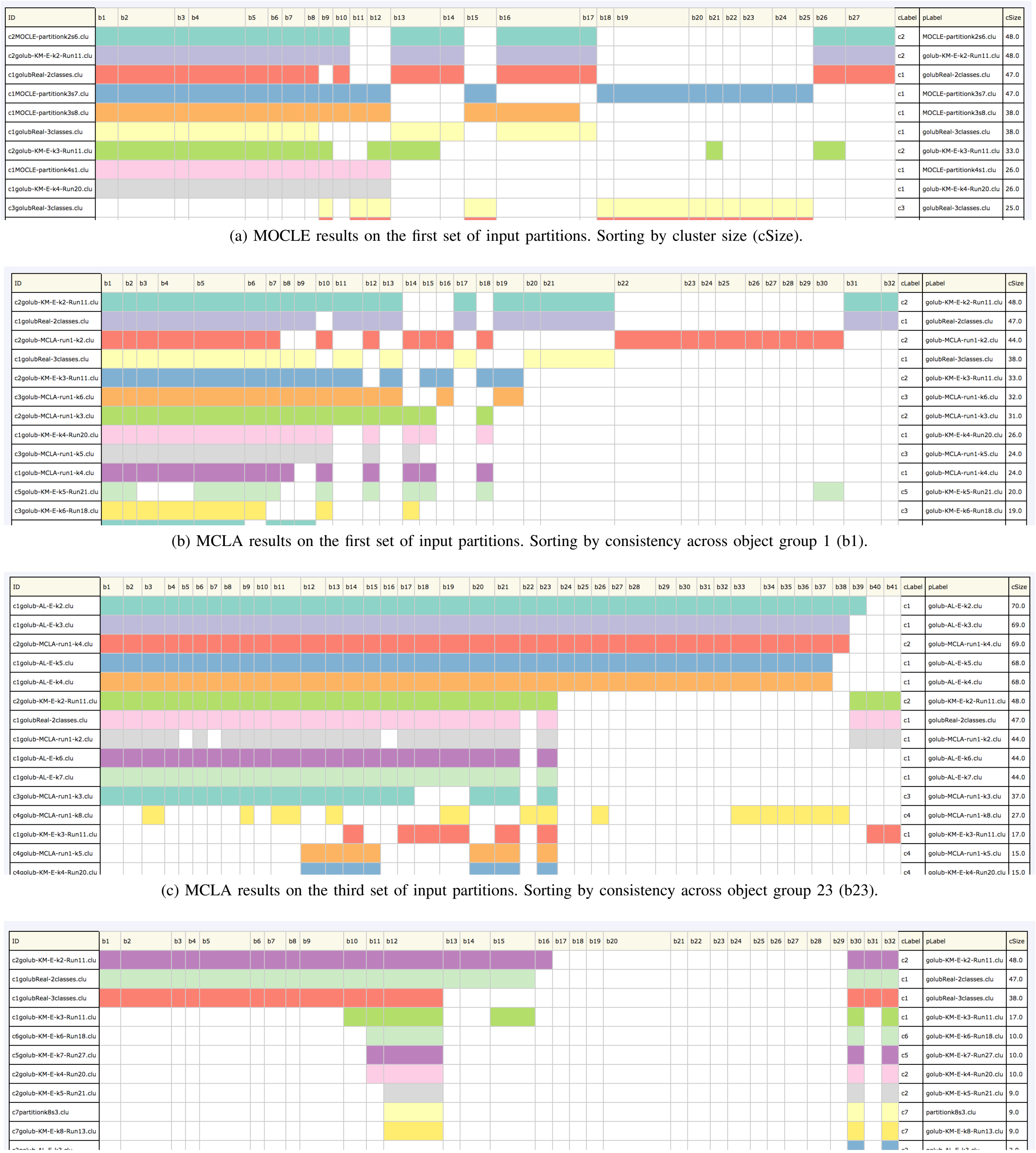
TABLE III
CHARACTERISTICS OF THE PARTITIONS USING BP-AL-K2-8 AS INPUT

Partition Set	TP	k	ARI
BP-KM-AL-k2-8	$k = 2$	2	0.943999
BP-KM-AL-k2-8	$k = 3$	8	0.798051
MOCLE	$k = 2$	6	0.876025
MOCLE	$k = 3$	8	0.798051
MCLA	$k = 2$	2	0.837557
MCLA	$k = 3$	2	0.654128

TABLE IV
CHARACTERISTICS OF THE PARTITIONS USING BP-KM-AL-K2-8 AS INPUT

Figure 2a provides a momentary snapshot of the analysis of MOCLE's results on the first collection of input partitions. A full interactive analysis can be conducted through http://lasid.sor.ufscar.br/visualization/cVis/cVis-BP-KM-k2-8---MOCLE-L4-MCLA-run1---TP.html. The set of KM input solutions contains a highly accurate solution, generated for $k = 2$. MOCLE is able to identify this partition, despite its underrepresentation in the original input set. An analysis using CVis clearly highlights the origin of MOCLE's performance in this single input solution. Furthermore, the visualization confirms that the information is fully retained within a single MOCLE solution only. Specifically, the snapshot included here displays one of the two clusters contained within the best MOCLE solution, in the first row. The second row highlights the origin of this cluster in the KM input solution. Finally, the third row confirms the excellent fit to one of the original classes in the Golub data. Where available, class information of this type may optionally be introduced into the analysis to support immediate inferences on cluster quality.

Figure 2b shows an extract of the analysis for



ID	cLabel	pLabel	cSize
c2MOCLE-partitionk2s6.clu	c2	MOCLE-partitionk2s6.clu	48.0
c2golub-KM-E-k2-Run11.clu	c2	golub-KM-E-k2-Run11.clu	48.0
c1golubReal-2classes.clu	c1	golubReal-2classes.clu	47.0
c1MOCLE-partitionk3s7.clu	c1	MOCLE-partitionk3s7.clu	47.0
c1MOCLE-partitionk3s8.clu	c1	MOCLE-partitionk3s8.clu	38.0
c1golubReal-3classes.clu	c1	golubReal-3classes.clu	38.0
c2golub-KM-E-k3-Run11.clu	c2	golub-KM-E-k3-Run11.clu	33.0
c1MOCLE-partitionk4s1.clu	c1	MOCLE-partitionk4s1.clu	26.0
c1golub-KM-E-k4-Run20.clu	c1	golub-KM-E-k4-Run20.clu	26.0
c3golubReal-3classes.clu	c3	golubReal-3classes.clu	25.0

(a) MOCLE results on the first set of input partitions. Sorting by cluster size (cSize).

ID	cLabel	pLabel	cSize
c2golub-KM-E-k2-Run11.clu	c2	golub-KM-E-k2-Run11.clu	48.0
c1golubReal-2classes.clu	c1	golubReal-2classes.clu	47.0
c2golub-MCLA-run1-k2.clu	c2	golub-MCLA-run1-k2.clu	44.0
c1golubReal-3classes.clu	c1	golubReal-3classes.clu	38.0
c2golub-KM-E-k3-Run11.clu	c2	golub-KM-E-k3-Run11.clu	33.0
c3golub-MCLA-run1-k6.clu	c3	golub-MCLA-run1-k6.clu	32.0
c2golub-MCLA-run1-k3.clu	c2	golub-MCLA-run1-k3.clu	31.0
c1golub-KM-E-k4-Run20.clu	c1	golub-KM-E-k4-Run20.clu	26.0
c3golub-MCLA-run1-k5.clu	c3	golub-MCLA-run1-k5.clu	24.0
c1golub-MCLA-run1-k4.clu	c1	golub-MCLA-run1-k4.clu	24.0
c5golub-KM-E-k5-Run21.clu	c5	golub-KM-E-k5-Run21.clu	20.0
c3golub-KM-E-k6-Run18.clu	c3	golub-KM-E-k6-Run18.clu	19.0

(b) MCLA results on the first set of input partitions. Sorting by consistency across object group 1 (b1).

ID	cLabel	pLabel	cSize
c1golub-AL-E-k2.clu	c1	golub-AL-E-k2.clu	70.0
c1golub-AL-E-k3.clu	c1	golub-AL-E-k3.clu	69.0
c2golub-MCLA-run1-k4.clu	c2	golub-MCLA-run1-k4.clu	69.0
c1golub-AL-E-k5.clu	c1	golub-AL-E-k5.clu	68.0
c1golub-AL-E-k4.clu	c1	golub-AL-E-k4.clu	68.0
c2golub-KM-E-k2-Run11.clu	c2	golub-KM-E-k2-Run11.clu	48.0
c1golubReal-2classes.clu	c1	golubReal-2classes.clu	47.0
c1golub-MCLA-run1-k2.clu	c1	golub-MCLA-run1-k2.clu	44.0
c1golub-AL-E-k6.clu	c1	golub-AL-E-k6.clu	44.0
c1golub-AL-E-k7.clu	c1	golub-AL-E-k7.clu	44.0
c3golub-MCLA-run1-k3.clu	c3	golub-MCLA-run1-k3.clu	37.0
c4golub-MCLA-run1-k8.clu	c4	golub-MCLA-run1-k8.clu	27.0
c1golub-KM-E-k3-Run11.clu	c1	golub-KM-E-k3-Run11.clu	17.0
c4golub-MCLA-run1-k5.clu	c4	golub-MCLA-run1-k5.clu	15.0
c4golub-KM-E-k4-Run20.clu	c4	golub-KM-E-k4-Run20.clu	15.0

(c) MCLA results on the third set of input partitions. Sorting by consistency across object group 23 (b23).

ID	cLabel	pLabel	cSize
c2golub-KM-E-k2-Run11.clu	c2	golub-KM-E-k2-Run11.clu	48.0
c1golubReal-2classes.clu	c1	golubReal-2classes.clu	47.0
c1golubReal-3classes.clu	c1	golubReal-3classes.clu	38.0
c1golub-KM-E-k3-Run11.clu	c1	golub-KM-E-k3-Run11.clu	17.0
c6golub-KM-E-k6-Run18.clu	c6	golub-KM-E-k6-Run18.clu	10.0
c5golub-KM-E-k7-Run27.clu	c5	golub-KM-E-k7-Run27.clu	10.0
c2golub-KM-E-k4-Run20.clu	c2	golub-KM-E-k4-Run20.clu	10.0
c2golub-KM-E-k5-Run21.clu	c2	golub-KM-E-k5-Run21.clu	9.0
c7partitionk8s3.clu	c7	partitionk8s3.clu	9.0
c7golub-KM-E-k8-Run13.clu	c7	golub-KM-E-k8-Run13.clu	9.0

(d) MOCLE results on the third set of input partitions. Sorting by consistency across object group 30 (b30).

Fig. 2. Snapshots of different orderings / selections for combined input and output collections of MCLA and MOCLE, on different input partitions.

MCLA's results on the same collection of input partitions. A full interactive analysis can be conducted through http://lasid.sor.ufscar.br/visualization/cVis/cVis-BP-KM-k2-8---MCLA-run1---TP.html. The tendency of KM (with high cluster numbers) to generate clusters that further segment the original, spatially sparated clusters is clearly apparent in the visualization. Shown here are the suggested KM divisions of one of the true clusters (as highlighted in the second row) for the KM $k = 2$ to $k = 5$ solutions. While the $k = 2$ solution (first row) is the highly accurate solution discussed above, this valuable information is diluted by the other KM solutions. The noisy nature of the input set causes problems for MCLA, and results in the generation of poor solutions that sub-divide the original clusters. In the figure, this becomes evident in terms of the discrepancies between the five MCLA solutions, as well as their lack of alignment with the true clustering solution.

Results on the second input set are comparable across methods and are not discussed further due to space limitations. Interactive results are available at http://lasid.sor.ufscar.br/visualization/cVis/cVis-BP-AL-k2-8---MOCLE-L4-MCLA-run1---TP.html and http://lasid.sor.ufscar.br/visualization/cVis/cVis-BP-AL-k2-8---MCLA-run1---TP.html.

Figure 2c provides a snapshot of the analysis for MCLA's results on the third collection of input partitions. A full interactive analysis can be conducted through http://lasid.sor.ufscar.br/visualization/cVis/cVis-BP-KM-AL-k2-8---MCLA-run1---TP.html. Using CVis, it becomes clear that MCLA mostly manages to handle the increase in noise caused by the addition of KM partitions to the AL inputs. Specifically, the AL partitions appear to carry sufficient information to support the identification of those cluster boundaries that occur in the partitions from both algorithms. In consequence, MCLA obtains a good solution for $k = 2$, and ordering by specific cluster boundaries of that partition (see Figure 2c) demonstrates that the specific support of these clusters arises from a combination of AL and KM partitions. MCLA's performance at identifying the true $k = 3$ partition is poorer. This is because the useful information carried by the AL partitions is diluted by the information carried by the KM solutions. Specifically, KM solutions disagree internally and there is no obvious agreement between the KM and AL solutions regarding the location of the additional cluster boundary.

Figure 2d shows part of the analysis for MOCLE's results on the same (third) input set. A full interactive analysis can be conducted through http://lasid.sor.ufscar.br/visualization/cVis/cVis-BP-KM-AL-k2-8---MOCLE-L4-MCLA-run1---TP.html. Interestingly, the number of object groups shown here is significantly smaller than in Figure 2c, reflecting the fact that MOCLE is strictly constrained by the cluster boundaries present in the input set. Like MCLA, MOCLE cannot take advantage of the union of two different types of input partitions. In particular, correspondence of final solutions to the true $k = 2$ partition is worse than it is for the first input data set (using KM partitions only), indicating that MOCLE is now unsuccessful in identifying the accurate (but under-represented) KM solution contained in this set. The visualization using CVis highlights that MOCLE's solutions do not manage to merge the singleton objects b30 to b32, which are part of the same class and are correctly identified by the $k = 2$ KM solution. The fact that the promising KM solution is not retained in the final population points to a failure of the clustering criteria to correctly differentiate between the quality of these partitions, for this particular data set.

V. Conclusion and Future Work

We have provided a first description of the visualization method CVis, which aims to improve insight into the relationships within sets of clustering solutions. This is of particular value in the context of ensemble techniques, which generate a set of ensemble clusterings from an input set of clustering solutions. We highlight specific strengths of CVis using the example of a gene expression data set.

State-of-the-art visualization tools draw their power from interactivity and there remain a number of ways in which interactivity in CVis can be improved further. CVis already provides a variety of mechanisms through which clusters can be sorted, but it does not yet allow for sorting by type of clustering method or cluster number (this can only be achieved implicitly by sorting through partition name). Sorting according to a ground truth or match to a highlighted cluster would also be helpful features.

Acknowledgment

This work is funded by grant#2015/21560-0, São Paulo Research Foundation (FAPESP) and grant EP/M013766/1, Engineering and Physical Sciences Research Council, UK.

References

[1] S. Vega-Pons and J. Ruiz-Shulcloper, "A survey of clustering ensemble algorithms," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 25, no. 03, pp. 337–372, 2011.

[2] P. Yang, Y. Hwa Yang, B. B Zhou, and A. Y Zomaya, "A review of ensemble methods in bioinformatics," *Current Bioinformatics*, vol. 5, no. 4, pp. 296–308, 2010.

[3] A. Strehl and J. Ghosh, "Cluster ensembles—a knowledge reuse framework for combining multiple partitions," *Journal of machine learning research*, vol. 3, no. Dec, pp. 583–617, 2002.

[4] J. Quackenbush, "Computational analysis of microarray data," *Nature reviews genetics*, vol. 2, no. 6, pp. 418–427, 2001.

[5] K. Faceli, M. C. P. Souto, D. S. A. de Araújo, and A. C. F. L. F. Carvalho, "Multi-objective clustering ensemble for gene expression data analysis," *Neurocomputing*, vol. 72, no. 13-15, pp. 2763–2774, 2009. [Online]. Available: http://dx.doi.org/10.1016/j.neucom.2008.09.025

[6] M. Delattre and P. Hansen, "Bicriterion cluster analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, no. 4, pp. 277–291, 1980.

[7] J. Handl and J. Knowles, "An evolutionary approach to multiobjective clustering," *IEEE transactions on Evolutionary Computation*, vol. 11, no. 1, pp. 56–76, 2007.

[8] T. R. Golub, P. T. D. K. Slonim and, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander, "Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring," *Science*, vol. 286, no. 5439, pp. 531–537, 1999.

[9] R. Xu and D. Wunsch, "Survey of clustering algorithms," *IEEE Transactions on Neural Networks*, vol. 16, no. 3, pp. 645–678, 2005.

[10] M. J. L. Hoon, S. Imoto, J. Nolan, and S. Miyano, "Open source clustering software," *Bioinformatics*, vol. 20, no. 9, pp. 1453–1454, 2004.

[11] K. Faceli, A. C. De Carvalho, and M. C. De Souto, "Multi-objective clustering ensemble," *International Journal of Hybrid Intelligent Systems*, vol. 4, no. 3, pp. 145–156, 2007.

[12] J. Handl, J. Knowles, and D. B. Kell, "Computational cluster validation in post-genomic data analysis," *Bioinformatics*, vol. 21, no. 15, pp. 3201–3212, 2005.