A First Study on Clustering Collections of Workflow Graphs

Emanuele Santos[1], Lauro Lins[1], James P. Ahrens[3], Juliana Freire[2], and Cláudio T. Silva[1,2]

[1] Scientific Computing and Imaging Institute, University of Utah
[2] School of Computing, University of Utah
[3] Los Alamos National Lab

Abstract. As workflow systems get more widely used, the number of workflows and the volume of provenance they generate has grown considerably. New tools and infrastructure are needed to allow users to interact with, reason about, and re-use this information. In this paper, we explore the use of clustering techniques to organize large collections of workflow and provenance graphs. We propose two different representations for these graphs and present an experimental evaluation, using a collection of 1,700 workflow graphs, where we study the trade-offs of these representations and the effectiveness of alternative clustering techniques.

1 Introduction

As workflow systems get more widely used, the number of workflows and the volume of provenance they generate has grown considerably. In fact, large collections of workflows have recently become available through Web sites that enable users to publish and share workflows [13, 19]. Yahoo! Pipes [19], for example, allows users to interactively create data mashups (represented as workflows) through a Web-based interface. Although Yahoo! Pipes has been online for a little over one year, there are already several thousand "pipes" stored on their servers.

The availability of large collections of workflows, such as the ones being held at workflow-sharing sites and in provenance repositories, creates new opportunities for exploring and mining this data. In this paper, we explore different techniques to cluster workflows. The ability to group similar workflows together has many important applications. For example, clustering can be used to automatically create a directory of workflows, similar to DMOZ (http://www.dmoz.org), that users can easily browse. Clustering can also be used to provide a better organization for search results. For example, Yahoo! Pipes provides basic search capabilities through keyword-based interfaces. But because the results are displayed as a long list, users have to go through the list and examine the results sequentially to identify the relevant ones. By clustering the results into distinct groups, users can have a more global and succinct view of the results and more quickly identify the information they are looking for.

The problem of clustering workflows, however, remains largely unexplored. This paper is, to the best of our knowledge, the first study on using clustering

J. Freire, D. Koop, and L. Moreau (Eds.): IPAW 2008, LNCS 5272, pp. 160–173, 2008.

techniques for workflow graphs. We explore different representations for these graphs as well as distance measures and clustering algorithms. We perform an experimental study, using a collection of 1,700 workflow graphs, where we examine the trade-offs of these configurations and the effectiveness of alternative clustering approaches.

The remainder of this paper is organized as follows. In Section 2, we review basic clustering concepts and discuss different choices for designing clustering strategies for workflows, including alternative representations for workflows and distance measures. We present our experimental evaluation in Section 3 and describe preliminary results that indicate that clustering strategies can be designed that are both scalable and effective for workflow graphs. We conclude in Section 4, where we outline directions for future work.

2 Clustering Workflows

Clustering is the partitioning of objects, observations or data items into groups (clusters) based on similarity. The goal is to group a collection of objects into clusters, such that the objects within each cluster are more related to one another than to those in different clusters.

Clustering techniques are widely applicable and have been used in many different areas (see [10] for a survey). These areas include, but are not limited to: document retrieval [2, 4], image segmentation [12, 18] and data mining [5]. Clustering has also been applied in the context of business workflows to derive workflow specifications from sequences of execution log entries [7].

To cluster a set of elements, three key components are needed: a model to represent the elements; a (dis)similarity measure or a distance metric; and a clustering algorithm. In this section we describe different alternatives for each of these components when the object of clustering is a workflow graph.

2.1 Alternative Workflow Representations

Data representation refers to the set of features that will be available to the clustering algorithm. A workflow can be defined as a network of tasks structured

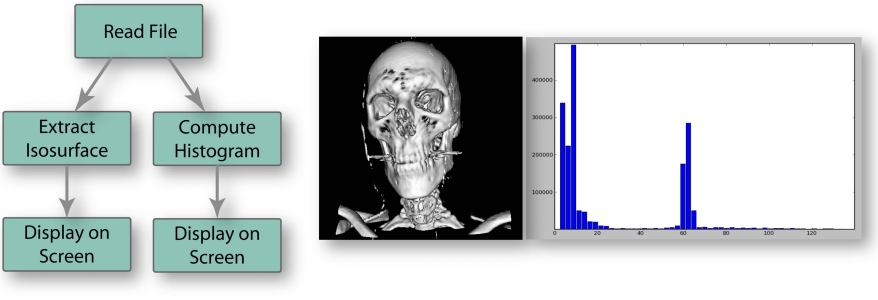

Fig. 1. On the left: a graph representation of a workflow in the visualization domain and on the right its generated data products

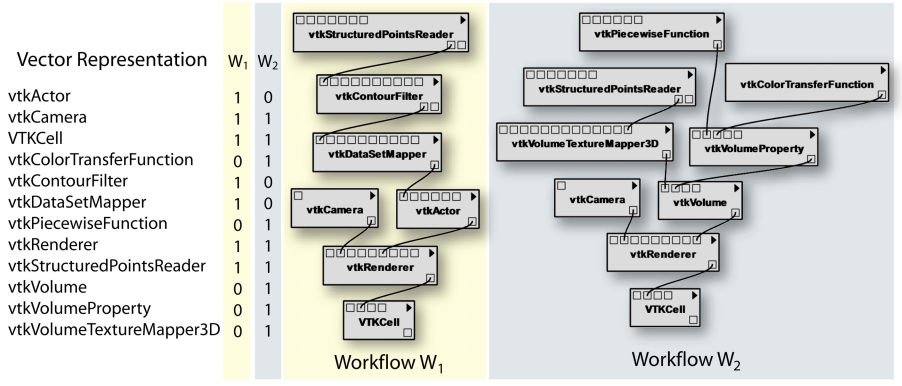

Fig. 2. Vector representation of two different VTK (Visualization Toolkit) workflows. The workflow on the left does isosurface extraction and the workflow on the right does volume rendering.

based on their control and data dependencies. Workflows can be represented as directed graphs, where nodes correspond to modules that perform computations and edges correspond to data streams, as shown on the left of Figure 1. For clustering purposes, we can select different features from these graphs. For example, a possible representation of this graph is to capture only the names of modules and the (unlabeled) edges between modules. More complex representations can be obtained if we take into consideration the parameter values and the input and output ports of each module.

For the clustering strategy to be effective, the data representation must include descriptive features in the input set (*feature selection*), or that new features based on the original set be generated (*feature extraction*). In the representation above, the selected features are the labeled workflow graphs, which is an example of a structured feature. Another way of representing a workflow is as a multidimensional vector [14], which is very popular in the information retrieval literature [1]. In our case, the dimensions in the vector space are defined by the union of all the possible module names the workflows in the input set may contain. Figure 2 illustrates the vector representation of two different workflows that combine modules from the Visualization Toolkit library (VTK) [11].

At first, this representation may seem not very suitable for workflows because the structural information is completely lost. However, we will see that representing workflows as vectors will have its advantages when we discuss similarity measures and clustering algorithms.

2.2 Measuring Workflow Similarity

The similarity measure is critical to any clustering technique and it must be chosen carefully. Usually, the similarity measure is a distance measure defined on the feature space. If we model workflows as graphs, graph-based distance measures can be used, such as edit distance [15], subgraph isomorphism [16],

and Maximum Common Induced Subgraph (MCIS). Consider for example MCIS. The distance measure d_{MCIS} derived from the MCIS of two graphs G_1 and G_2 is defined as [3]:

$$d_{\text{MCIS}}(G_1, G_2) = 1 - \frac{|\text{mcis}(G_1, G_2)|}{\max\{|G_1|, |G_2|\}}$$

Intuitively, the larger a MCIS of two graphs is, the more similar the two graphs are. Notice that if two graphs are isomorphic, their MCIS distance is 0 and if they do not have any common subgraph, their MCIS distance is 1. Bunke and Shearer [3] also demonstrated that the MCIS distance satisfies the properties of a metric. The problem with this measure is that it is computationally expensive and for a large collection of workflows, that can be a limitation.

When workflows are represented using the vector space (VS) model, other distance metrics can be used (e.g., Euclidean and Euclidean squared distances). A widely-used distance metric for VS is the cosine distance d_{VS} between two vectors v_1 and v_2, defined as:

$$d_{\text{VS}}(v_1, v_2) = 1 - \cos\theta = 1 - \frac{v_1 \cdot v_2}{\|v_1\|\|v_2\|}$$

Figure 3 shows a concrete example of how d_{MCIS} and d_{VS} are computed for two structurally different graphs. Note their different behaviors: while MCIS is able to capture the (structural) difference between the workflows, the cosine distance is not. This example highlights the importance of selecting an appropriate representation and distance measure.

The input set can be represented directly in terms of the dissimilarity between pairs of observations. This can be done by means of a matrix of dissimilarities, which is a $N \times N$ matrix M, where N is the number of observations and each element m_{ij} contains the distance between observations i and j.

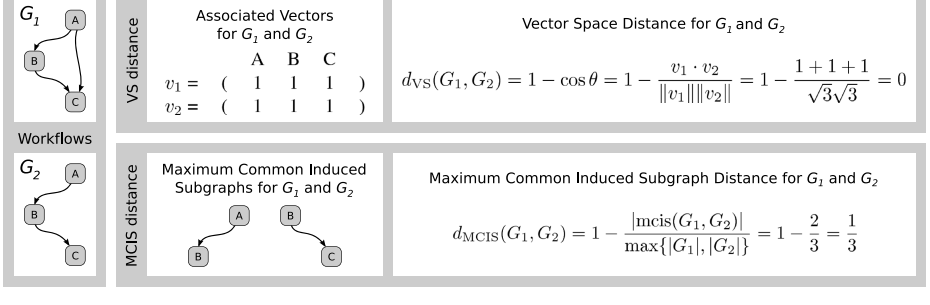

Fig. 3. Vector Space (VS) distance and Maximum Common Induced Subgraph (MCIS) distance for workflows G_1 and G_2. Notice that the VS distance does not capture structural differences (i.e., VS distance equals zero) and that although the path $A \rightarrow B \rightarrow C$ is a common subgraph of G_1 and G_2, it is not an induced subgraph of G_1.

2.3 Clustering Algorithms

There are many different approaches to clustering data. Roughly speaking, the cluster algorithms can be classified as hierarchical or partitioning (see [10] for a more comprehensive taxonomy of clustering techniques). Partitioning algorithms produce only a single partition of the input set while hierarchical methods produce a nested series of partitions. One of the most popular partitioning methods is the K-means algorithm. K-means partitions the input set N into K clusters in such a way that it minimizes the intracluster dissimilarity or equivalently maximizes the intercluster dissimilarity [8]. Intracluster dissimilarity D_{intra} is defined as:

$$D_{intra} = \frac{1}{2} \sum_{k=1}^{K} \sum_{m \in k} \sum_{n \neq m \in k} d(x_m, x_n)$$

and intercluster dissimilarity D_{inter} is defined as:

$$D_{inter} = \frac{1}{2} \sum_{k=1}^{K} \sum_{m \in k} \sum_{n \in k' \neq k} d(x_m, x_n)$$

Summing both dissimilarities, we obtain the total point scatter T of the input set, which is independent of cluster assignment:

$$T = D_{inter} + D_{intra} = \frac{1}{2} \sum_{m=1}^{N} \sum_{n=1}^{N} d(x_m, x_n)$$

Because it is not practical to compute this by exhaustive enumeration, K-means works in a iterative greedy descent fashion, as described below:

1. Specify the initial K cluster centers
2. Assign each observation to the closest center
3. Recompute centers of each cluster as the mean of the observations in the cluster
4. If assignments have changed, go to 2.

The problem with K-means is that computing centers is possible only with the vector space based features. In order to work with arbitrary representations, such as given by a matrix of dissimilarities, the algorithm can be generalized to the K-medoids algorithm, in which at each iteration the centers are restricted to be one of the observations assigned to the cluster. The cost of performing K-means is proportional to KN and the cost of performing K-medoids is proportional to KN^2, which is computationally more expensive.

The advantage of these methods is that they converge rather quickly and are very easy to implement. The disadvantages of both K-means and K-medoids are the choice of the parameter K and the fact that they are very sensitive to the initialization. Because of that we often need to run these algorithms a few times in order to get the best cluster configuration. Another problem is that they do

not present an order relation inside each cluster, and when this is important, using a hierarchical clustering technique is a better option.

Hierarchical clustering algorithms, as their name suggests, build hierarchical representations such that the clusters at each level of the hierarchy are formed by merging two clusters at the next lower level. So, at the lowest level, each cluster has a single object and at the highest level, there is only one cluster containing all the objects. Then, there are $N-1$ levels in the hierarchy. Hierarchical methods require neither an initialization nor a parameter K. However, they do require the specification of a dissimilarity measure between groups of objects, based on the pairwise dissimilarities among the objects in the two groups.

Depending on the strategy chosen to build the hierarchy, the algorithms can be classified as agglomerative (bottom-up) or divisive (top-down) [8]. In the agglomerative approach, the process is started at the bottom, and recursively at each level two clusters with the smallest intercluster dissimilarity are merged to form the next level, which will have one less cluster. Divisive approaches, on the other hand, start at the top and recursively at each level a cluster is divided into two new clusters such that they present the largest intercluster dissimilarity. These recursive processes can be represented by a rooted binary tree. Figure 7 illustrates the results of running K-medoids on an input set containing 50 workflows, using the two dissimilarities measures described above. The last column of the spreadsheet on the left shows the agglomerative representation for each dissimilarity measure.

3 Experimental Evaluation

Our goal in this experimental evaluation is to assess the effectiveness of different approaches to clustering workflows. In particular, we study the trade-offs between a graph-based and a vector-based representation for workflows, and compare different clustering algorithms. Before discussing our results, below we describe the dataset we used in the experiments.

3.1 The Dataset

The workflows used in this study were generated by thirty students during a scientific visualization course. Over the semester, the students were asked to design workflows to solve different visualization problems (e.g., generate an isosurface visualization of a skull or create a vector field visualization of the salinity of a river). All these tasks were performed in VisTrails [17], a workflow development tool that captures provenance of workflow evolution [6], i.e., all refinements and parameter explorations performed by users during workflow design. For each assignment, the students turned in a file containing detailed provenance of their work, including all different workflow variations they created to solve the problems in the assignment. They were instructed to tag the actual solution workflows with a meaningful label, so that these could be (easily) identified by the instructor and TAs.

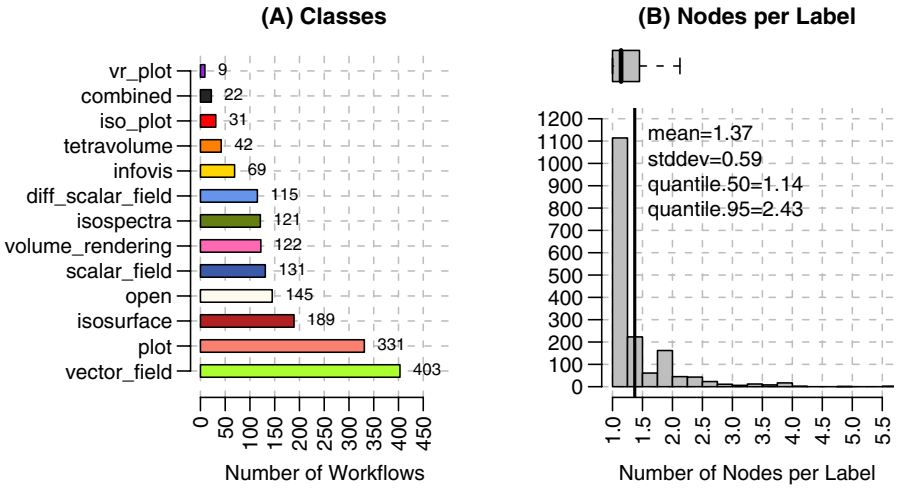

Fig. 4. (A) The initial 13 classes used to partition $\mathcal{W}$ and the number of workflows in each class. (B) Box-plot, histogram and some statistics for the distribution of the number of nodes per number of labels in $\mathcal{W}$.

To assemble our dataset $\mathcal{W}$ we extracted from the files only the workflows identified with a "solution" tag: a total of 1730 workflows. We also classified these workflows, so that we could have reference data to evaluate the quality of the resulting clusters. The classification was done as follows. Based on the assignment problem and the tag provided by the student, we classified each workflow by the type of problem they were supposedly solving. For example, a workflow for assignment 1 tagged as `Problem 1` was classified as `isosurface`, since problem 1 in this assignment asked the students design a workflow for extracting isosurfaces of a 3-dimensional object. For some problems, a specific technique was not required, the workflows created for these problems were classified as `open`. Figure 4 (A) shows the 13 classes we used and the number of workflows in each class.

The workflows in $\mathcal{W}$ contained all information necessary for running them: modules, connections, dependencies, parameters, parameter values, etc. For clustering purposes we use a simplified representation for the workflows that preserves only the module names and connection information: we abstract a workflow as a directed simple labeled graph. More formally, a workflow W is a triple $W = (N, A, \ell : N \to L)$, where N is the set of modules or *nodes*, A is the set of *arcs*, which is a subset of all ordered pairs in $N \times N$ and ℓ is a function assigning one *label* in the set L to each node in N. Simple graphs have no loops, so pairs (x, x) are not allowed in A.

Although $\mathcal{W}$ contained 1730 workflows some of their graph representations were exactly the same (i.e., *isomorphic graphs*). This was expected to happen since many workflows in $\mathcal{W}$ were designed to solve the same problem. So, for our

purposes, instead of using $\mathcal{W}$ we used its subset $\mathcal{W}'$ that consisted of the 1031 different (i.e., *non-isomorphic*) graphs in $\mathcal{W}$.

3.2 Deriving Clusters

Based on the workflow abstraction described in Section 3.1, we used the representations, the dissimilarity measures and the algorithms detailed in Section 2 to cluster the workflows in $\mathcal{W}$. Throughout this section we will use the term MCIS to refer to the structural representation and dissimilarity configuration and VS to refer to the vector-space and cosine distance configuration.

We constructed two distance matrices M'_{mcis} and M'_{vs} for $\mathcal{W}'$ based on the MCIS and VS distance measures. These matrices were used as inputs for the clustering algorithms we experimented with: K-medoids and hierarchical (agglomerative) clustering algorithms were used for both VS and MCIS; and K-means was applied to the VS configuration.

For K-medoids and K-means, to select an appropriate value for K, each configuration was executed 50 times for each specific value of K, with K varying from 2 to 20. For each execution we computed the D_{intra} and D_{inter} cluster dissimilarities and picked the best values, which for the final results were $K = 8$ in the VS configuration and $K = 9$ in the MCIS configuration. The criterion used for choosing the values of K is illustrated in Figure 7, which shows its usage in preliminary results: we examine the values of $logD_{intra}$ as a function of the number of clusters K and search for a "kink" in the plot to choose the most interesting values of K for both configurations [8].

3.3 Effectiveness of Clustering

By examining visualizations of the clustering results, including the ones shown in Figures 5 and 6, we can observe that, for the most part, workflows that belong to the same class are grouped together for both VS and MCIS configurations. There are, however, classes that are spread out across (many) different clusters. As Figure 5 shows, most workflows in the `vector_field` and `infovis` classes are grouped in in the first and second clusters (the first two bars, starting from the bottom). However, workflows classified as being `vector_field` are also found in other clusters.

While trying to understand the heterogeneity of some of the clusters, we came across an interesting and unexpected finding: our classification based on assignment problem and student-specified tag was not accurate for all classes. We selected some of the workflows classified as `vector_field` but that ended up in different clusters (A and B)—which we refer to as `vector_field1` and `vector_field2`. We also selected two workflows in cluster A which belong to different classes: `vector_field1` and `isospectra`. Then, we compared them, side-by-side. The visual difference results for the two workflow pairs, displayed in Figure 6, show that: `vector_field1` and `vector_field2` have no modules in common; and `vector_field1` and `isospectra` have a very similar structure, which differs in a single module. The workflows were actually correctly grouped.

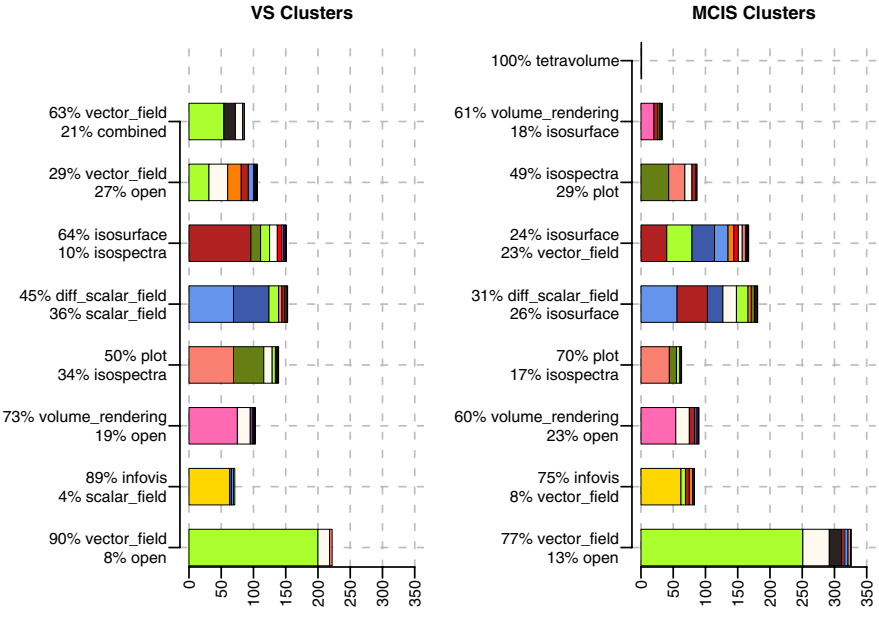

Fig. 5. Clustering $\mathcal{W}'$ using the VS and MCIS distances. The percentages of the two "initial" classes (see Figure 4(A)) that had the most number of workflows inside each cluster are reported. The bars were manually ordered trying to align similar color patterns. The colormap is also the same as used in Figure 4(A). Notice how the first 4 bars (bottom to top) present a similar pattern. *This figure is best understood if viewed in color.*

This indicates that clustering can be an effective means to organize workflow collections.

Although the results produced by K-medoids give some insight into the different types of workflows in our dataset, they do not provide much information about the relationship between workflows in each cluster. To understand these relationships, we used an agglomerative representation to inspect the behavior of both distance measures in more detail. Figure 7 shows, side by side, the results from MCIS and VS using K-medoids and agglomerative clustering. The relationship between the workflows in a cluster are easily seen by looking at the structure of the agglomerative trees. Interesting observations can be drawn from these trees. Notice in both trees that there is a cluster with a single observation (stemming from the root): they correspond to the same workflow. This workflow is an outlier because it contains a single module that does not appear frequently in other workflows in our dataset.

This hierarchical representation can also help in the selection of an appropriate value for K. Depending on the distance metric used, the best values for K can be different. When running K-medoids on a subset of $\mathcal{W}$ containing 50 workflows, $K = 6$ was chosen for MCIS and $K = 4$ for VS. These values are highlighted

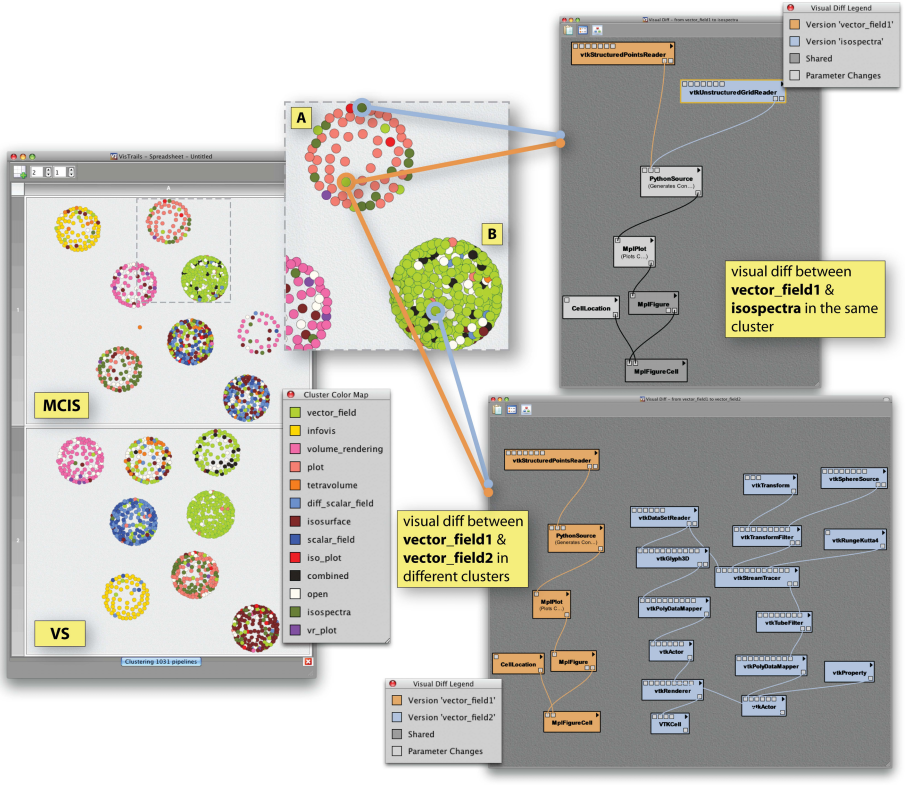

Fig. 6. Clustering results for 1031 workflows. On the spreadsheet (left) are the results of K-Medoids for MCIS (K=9) and for VS (K=8). The visual difference between representative workflows are shown on the right. They explain why observations classified as `vector_field` are in different clusters and why `isospectra` and `vector_field` observations were assigned to the same cluster. *This figure is best understood if viewed in color.*

in the plots on the right of the figure. Note that the both hierarchies in the figure have a number of subtrees that is similar to the K we selected for each configuration.

3.4 Workflow Representations: Graphs vs. Vectors

Figures 5 and 6 show an interesting pattern: the different representations and associated distance measures lead to similar clusters. Consider for example, the first four bars (bottom-up) of the two solutions in Figure 5 have a similar color pattern and size. Given that one representation captures the graph structure and the other is completely unstructured (i.e., it considers a workflow as a bag of words), this result was surprising to us.

To compare in more detail the graph-based and vector-based representations for workflows, we plotted the values of the distance matrices M'_{mcis} and M'_{vs}.

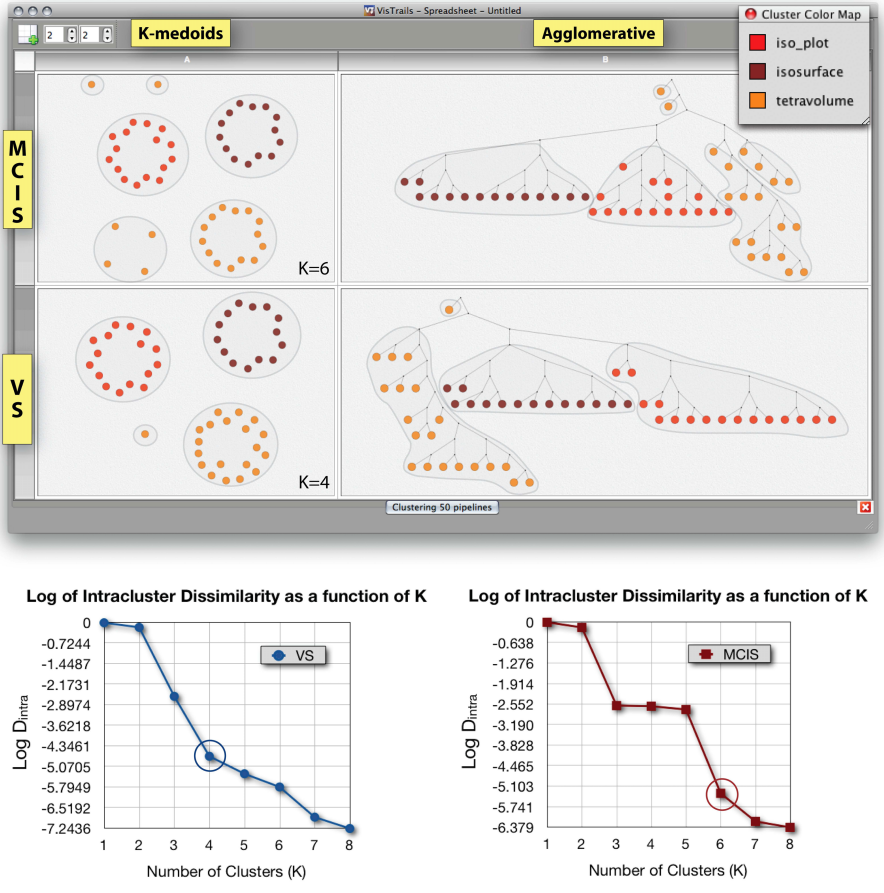

Fig. 7. Clustering results for 50 workflows. The groups formed by K-medoids are indicated in the agglomerative views. The plots below the spreadsheet show log D_{intra} as a function of the number of clusters (K) for each measure, where the chosen values of K are highlighted. The curves were translated to 0 at K=1.

Figure 8 shows a plot of the values in these matrices. Notice that the plot of the MCIS distances does not start from zero. This happens because the d_{mcis} is zero only if it is applied to a pair of isomorphic graphs and by construction there are no such pairs in $\mathcal{W}'$. The same does not occur to the VS plot: d_{vs} can be zero even when the graphs are different (see Figure 3 for an example). Note that this plot shows that the distances capture by these two distinct measures are similar.

We also compared the clusters produced by the two configurations: we used the Jaccard similarity coefficient [9], which is a well-known index for comparing two partitions of the same set. The larger this number is, the more similar the partitions are. Let $C_{vs}^{K=8}$ and $C_{mcis}^{K=9}$ denote the clustering results produced by the VS and the MCIS configurations, respectively. The Jaccard index for

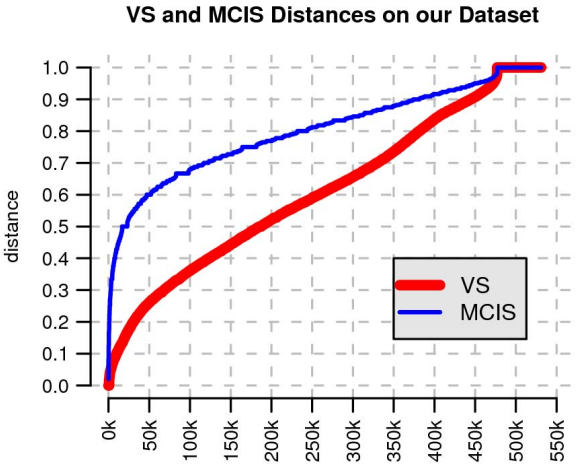

Fig. 8. For the 1031 workflows of $\mathcal{W}'$ we computed 530965 (= $1031 \times 1030/2$) VS and MCIS distance values. Ordering (independently) all these values for the two distance measures resulted in the above plot.

partitions $C_{vs}^{K=8}$ and $C_{mcis}^{K=9}$ was 0.328. To better understand what this number means, we checked if a partition of $\mathcal{W}'$ that matched $C_{mcis}^{K=9}$ as well as $C_{vs}^{K=8}$ could be found by chance. We then computed the Jaccard index between $C_{mcis}^{K=9}$ and 1000 randomly generated partitions of $\mathcal{W}'$, with $K = 9$. The mean value of the Jaccard index on this experiment was 0.08 and the maximum value was 0.082, very distant from the number obtained for the MCIS and VS clusters. This supports our hypothesis that the VS and MCIS configurations are correlated.

These results suggest that labels in $\mathcal{W}'$, the only information used by VS, capture a certain amount of the graph structure. To gain insight into this, we examined the distribution of labels across workflows (see Figure 4 (B)). A label appears in 1.37 workflows on average, with a standard deviation of 0.59. Thus, for our dataset, the number of labels is a good estimation to the number of nodes in a workflow (e.g., in 50% of our workflows the number of nodes was at most 1.14 times the number of labels). Also, empirically, we have observed that for the workflows in our dataset, the number of possible connections between modules is small, and it is constrained by the module labels. Intuitively, there is a large number of module pairs, but very few are compatible and can be directly connected.

4 Conclusion

We have presented a first study on clustering workflow graphs. We explored different representations for these graphs, studied the trade-offs of these representations, and assessed the effectiveness of alternative clustering techniques. Our experimental results show that clustering can be effective to organize large

collections of workflows. We have also observed that for our dataset, using a vector-space based representation produced good results—comparable to results obtained using the more costly structural representation.

There are several directions we plan to pursue in future work. Although our preliminary results suggest that, for our dataset, the vector space representation for workflows can be a cost-effective and scalable strategy to cluster large collections, additional experiments are needed to verify whether a similar behavior is obtained in other workflow collections. We also plan to investigate more systematic methods to determine the value of K (for K-medoids and K-means) as well as experiment with more complex representations of workflows, for instance, that capture parameter values and information about input and output ports for the modules.

Acknowledgments

Our research has been funded by the Department of Energy SciDAC (VACET and SDM centers), the National Science Foundation (grants IIS-0746500, CNS-0751152, IIS-0713637, OCE-0424602, IIS-0534628, CNS-0514485, IIS-0513692, CNS-0524096, CCF-0401498, OISE-0405402, CCF-0528201, CNS-0551724), and IBM Faculty Awards (2005, 2006, 2007, and 2008). E. Santos is partially supported by a CAPES/Fulbright fellowship.

References

1. Baeza-Yates, R.A., Ribeiro-Neto, B.A.: Modern Information Retrieval. ACM Press/Addison-Wesley (1999)
2. Barbosa, L., Freire, J., da Silva, A.S.: Organizing hidden-web databases by clustering visible web documents. In: Proceedings of the 23rd International Conference on Data Engineering, ICDE 2007, pp. 326–335. IEEE, Los Alamitos (2007)
3. Bunke, H., Shearer, K.: A graph distance metric based on the maximal common subgraph. Pattern Recognition Letters 19(3-4), 255–259 (1998)
4. Cutting, D.R., Karger, D.R., Pedersen, J.O., Tukey, J.W.: Scatter/gather: a cluster-based approach to browsing large document collections. In: SIGIR 1992: Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval, pp. 318–329 (1992)
5. Ester, M., Frommelt, A., Kriegel, H.-P., Sander, J.: Spatial data mining: Database primitives, algorithms and efficient dbms support. Data Mining and Knowledge Discovery 4(2-3), 193–216 (2000)
6. Freire, J., Silva, C.T., Callahan, S.P., Santos, E., Scheidegger, C.E., Vo, H.T.: Managing rapidly-evolving scientific workflows. In: Moreau, L., Foster, I. (eds.) IPAW 2006. LNCS, vol. 4145, pp. 10–18. Springer, Heidelberg (2006)
7. Greco, G., Guzzo, A., Pontieri, L., Sacca, D.: Discovering expressive process models by clustering log traces. IEEE Transactions on Knowledge and Data Engineering 18(8), 1010–1027 (2006)
8. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning: Data Mining, Inference and Prediction. Springer Series in Statistics. Springer, Heidelberg (2001)

9. Jaccard, P.: Étude comparative de la distribution florale dans une portion des Alpes et des Jura. Bulletin del la Société Vaudoise des Sciences Naturelles 37, 547–579 (1901)
10. Jain, A.K., Murty, M.N., Flynn, P.J.: Data clustering: a review. ACM Computing Surveys 31(3), 264–323 (1999)
11. Kitware. The Visualization Toolkit (March 15, 2008), http://www.vtk.org
12. Makrogiannis, S., Economou, G., Fotopoulos, S., Bourbakis, N.: Segmentation of color images using multiscale clustering and graph theoretic region synthesis. IEEE Transactions on Systems, Man and Cybernetics, Part A 35(2), 224–238 (2005)
13. MyExperiment (March 15, 2008), http://myexperiment.org
14. Salton, G., Wong, A., Yang, C.S.: A vector space model for automatic indexing. Communications of ACM 18(11), 613–620 (1975)
15. Sanfeliu, A., Fu, K.S.: A distance measure between attributed relational graphs for pattern recognition. IEEE Transactions on Systems, Man and Cybernetics (Part B) 13(3), 353–363 (1983)
16. Ullmann, J.R.: An algorithm for subgraph isomorphism. J. ACM 23(1), 31–42 (1976)
17. The VisTrails Project (March 15, 2008), http://www.vistrails.org
18. Wu, Z., Leahy, R.: An optimal graph theoretic approach to data clustering: theory and its application to image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence 15(11), 1101–1113 (1993)
19. Yahoo! Pipes (March 15, 2008), http://pipes.yahoo.com