14e édition des Journées francophones

EGC 2014

Extraction et Gestion des Connaissances

28-31 Janvier

RENNES

IRISA & Centre Inria Rennes - Bretagne Atlantique

Campus de Beaulieu, Rennes

Journées Ateliers/Tutoriels

VIF : Visualisation d'informations, interactions et fouille de données

Atelier visualisation d'informations, interaction et fouille de données

Organisateurs : : Hanene Azzag (LIPN, Université de Paris 13),
Fatma Bouali (LI, Université François-Rabelais Tours et Université de Lille2),
Fabien Picarougne (LINA, Université de Nantes),
Bruno Pinaud (LABRI, Université de Bordeaux),
Gilles Venturini (LI, Université Université François-Rabelais Tours)

PRÉFACE

L'édition 2014 de l'atelier "Visualisation d'informations, interaction et fouille de données" a pour but de fournir une présentation de méthodes nouvelles, d'axes de recherche, de développements et d'applications dans les domaines de la visualisation d'informations, de la fouille visuelle de données et plus généralement des approches visuelles et interactives pour la représentation et l'extraction d'informations et de connaissances.

Cet atelier prend pour support le groupe de travail Visualisation d'informations, interaction et fouille de données (GT-VIF). Notre ambition est de permettre aux participants d'aborder tous les thèmes de la visualisation et concerne aussi bien les chercheurs du monde académique que ceux du secteur industriel, et autant les notions conceptuelles que les applications. Les thèmes abordés en visualisation sont très nombreux et nous nous intéresserons principalement aux questions et aux problématiques liées aux grands volumes de données (les "Big Data"), comme la définition de nouvelles visualisations, l'utilisation du parallélisme (Cloud, CPU, GPU) ; à la "Business Intelligence", l'OLAP et autres bases de données ; et à l'Open et Linked data, qui soulèvent des problématiques très variées.

Nous débattrons également sur les succès et échecs récemment rencontrés.

Hanene Azzag
LIPN, Université de Paris 13

Fatma Bouali
LI, Université François-Rabelais Tours
et Université de Lille

Fabien Picarougne
LINA, Université de Nantes

Bruno Pinaud
LABRI, Université de Bordeaux

Gilles Venturini
LI, Université Université François-Rabelais Tours

Membres du comité de lecture

Le Comité de Lecture est constitué de:

Hanene Azzag (LIPN, Université de Paris 13)
Sadok Ben Yahia (Université de Tunis El Manar, Tunisie)
Fatma Bouali (LI Tours et Université de Lille2)
Romain Bourqui (LABRI, Université de Bordeaux)
Jean-Daniel Fekete (INRIA)
Fabrice Guillet (LINA, Université de Nantes)
Pascale Kuntz (LINA, Université de Nantes)
Mustapha Lebbah (LIPN, Université de Paris 13)
Guy Mélançon (LABRI, Université de Bordeaux)
Monique Noirhomme (Institut d'Informatique, FUNDP, Namur, Belgique)
Benoit Otjacques (Centre Gabriel Lippmann, Luxembourg)
Fabien Picarougne (LINA, Université de Nantes)
Bruno Pinaud (LABRI, Université de Bordeaux)
François Queyroi (LABRI, Université de Bordeaux)
Paul Richard (ISTIA, Université d'Angers)
Gilles Venturini (LI, Université de Tours)

TABLE DES MATIÈRES

Partie 1

How to visualize Information by using the notion of Memory Islands (demonstration)
Bin Yang et Jean-Gabriel Ganascia 1

Self-organizing Trees for visualizing protein dataset
Nhat-Quang Doan, Hanane Azzag, Mustapha Lebbah et Guillaume Santini 3

Construction et exploration d'un graphe de proximité portant sur une grande collection de jeux de données publiques
Tianyang Liu, Fatma Bouali et Gilles Venturini 21

Démonstration d'une visualisation radiale pour l'exploration de grands volumes de données
Tianyang Liu, Fatma Bouali et Gilles Venturini 29

Traitement Visuel et Interactif dans le Logiciel Cadral (démonstration)
Philippe Pinheiro, Yoann Didry, Olivier Parisot et Thomas Tamisier 33

Un système de clustering visuel et interactif
Pierrick Bruneau, Philippe Pinheiro, Bertjan Broeksema et Benoît Otjacques 45

Index des auteurs **49**

How to Visualize Information by using the notion of Memory Islands

Bin Yang[*,**], Jean-Gabriel Ganascia[*,**]

[*]Université Pierre et Marie Curie - Sorbonne Universités (Paris VI University)
[**]Labex OBVIL, ACASA team Laboratoire d'Informatique de Paris 6 (LIP6)
FirstName.LastName@lip6.fr

Résumé. Le terme « îles des mémoires » a été inspiré par l'ancien « Art de la mémoire», qui décrit comment dans l'antiquité et au Moyen Age la spatialisation était utilisée pour augmenter la capacité de mémorisation des lecteurs. La méthode des «loci» (pluriel de « locus » en latin qui signifie « endroit » ou « lieu ») consiste à créer une carte virtuelle et à associer chaque entité à des zones désignées sur la carte. Dans cet exposé, nous présenterons brièvement une nouvelle méthode dans le domaine de la visualisation cartographique des connaissances (par exemple une ontologie et son squelette qui est une taxonomie) basée sur la notion d'« îles des mémoires ». Nous avons déjà commencé le développement d'un outil de cartographie de contenus fondé sur l'idée d'« îles des mémoires » présentée ci-dessus. Nous présentons ensuite quelques exemples avec différentes connaissances hiérarchiques pour montrer comment la technique des «îles des mémoires» permet de naviguer dans les informations pour mémoriser leurs emplacements et les récupérer. Enfin, nous présentons un prototype expérimental destiné à évaluer la pertinence psychologique des îles de mémoires. Nous discutons également des résultats encourageants obtenus par une évaluation empirique préliminaire, qui montre que l'utilisation d'île de mémoire offre des avantages pour les utilisateurs non expérimentés s'attaquant à une navigation réaliste.

1 Datasets

- Knowledge data : Ontologies on line ;
  - InPhO Ontology : https ://inpho.cogs.indiana.edu/
  - Other ontologies.
- Other non-numerical Information ;
  - EBook : oedipe le maudit (of SEJER),
  - Public discussion,
  - Other information who need to visualized : e.g. http ://knowescape.org.

2 Jeux de données

- Données de connaissances : ontologies en ligne ;
  - InPhO Ontologie : https ://inpho.cogs.indiana.edu/
  - autre ontologies.
- Autre type d'Informations non-numérique ;
  - EBook : oedipe le maudit (de SEJER),
  - Le débat public,
  - Autres informations qui ont besoin d'être visualisées : par exemple, http ://knowescape.org/

Références

Yang, B. et J.-G. Ganascia (2013). Memory islands : an approach to cartographic visualization. In A. Slavic, A. Salah, et S. Davies (Eds.), *Proc. Classification and visualization : interfaces to knowledge : proceedings of the International UDC Seminar 24-25 October 2013, The Hague, The Netherlands*, Wurzburg, Germany. Ergon Verlag.

Annexe

Some examples can be found in our site : http ://www-poleia.lip6.fr/~polyle/

Summary

The term "Memory Islands" was inspired by the ancient "Art of Memory" which described how people in the antiquity and the Middle-Ages used spatialization to increase their memory capacity. The method of "loci" (plural of Latin locus for place or location) consists of creating a virtual map and associating each entity to designated areas on the map. In this talk, we will briefly present our new information visualization method based on the notion of Memory Islands for hierarchical knowledge (such as ontologies). We first describe our novel method for cartographic visualization of knowledge (e.g. ontology and its skeleton which is taxonomy); we then present some examples with different hierarchical knowledge to show how the technique of "Memory Islands" helps to navigate through information contents to memorize their locations and to retrieve them. Finally, we present an experimental prototype that is intended to evaluate the psychological relevancy of Memory Islands. We present also some preliminary empirical results showing that the use of Memory Islands provides advantages for non-experienced users tackling realistic browsing and visualization tasks.

Self-organizing Trees for visualizing protein dataset

Nhat-Quang Doan*, Hanane Azzag*, Mustapha Lebbah *, Guillaume Santini *

* Université Paris 13, LIPN UMR 7030
99, avenue Jean-Baptiste Clément
93430 Villetaneuse, France
nhat-quang.doan, hanane.azzag, lebbah,Santini@lipn.univ-paris13.fr

Résumé. Clustering and visualizing multi-dimensional or structured data are important tasks for data analysis, especially in bioinformatics. Self-organizing models are often used to address both of these issues. In this paper we introduce a hierarchical and topological visualization technique called Self-organizing Trees (SoT) which is able to represent data in hierarchical and topological structure. The experiment is conducted on a real-world protein data set.

1 Introduction

In bioinformatics, visualizing high-dimensional and large-scale biological data sets such as protein or gene expression data is a challenge. Proteins receive attention from experts because their structures harbor information that is not immediately obvious without integration and analysis. Graphical representations help experts to analyze and interpret easily protein data sets. The Protein Data Bank (Bernstein et al., 1977) (PDB) offers a comprehensive view of the available three-dimensional (3D) protein structures. In order to investigate their evolutionary relationships by detecting shared similarities at the sequence and at the structure level, a first step is to establish a classification of these objects. As for all complex objects, the clustering process is a difficult task. It can be done automatically by an algorithm, *manually* by a human expert or by combining automated and manual approaches. So far, reference protein structure classifications used in structural biology like SCOP (Murzin et al., 1995) or CATH (Pearl et al., 2003) are completely or partially constructed by human experts. If they present the advantage of being of high quality, the counterpart is the difficulty to maintain their exhaustivity in a context of quadratic growth of solved 3D-structures. For example, the last release of SCOP classification (Andreeva et al., 2008) (version 1.75) done in june 2009 do not take into account PDB entries added from that date which represent more than 25000 structures and a growth of about 40% between 2009 and 2012. To face with this issue a first solution consist in the development of more accurate algorithmic tools to cluster or classify automatically 3D biological molecules. Another way is to propose to human experts new solutions to help in the investigation of protein structure similarities.

In this work we propose an original visualization tool to investigate protein domain structure similarity proximities. First, a pairwise alignment of all proteins is done. Then a similarity graph G is built. Vertices of G represent protein structures and an edge occurs between two proteins when they share a *significant* similarity. Even if this graph is relatively sparse, its

visualization remains largely useless as the edge density remains too high for a human eye. We propose to simplify this representation using an algorithm based on Self- Organizing Map (SOM) called SoT (Doan et al., 2012).

SOM algorithm (Kohonen, 2001) is a popular unsupervised clustering method which is able to preserve data topology by using certain neighborhood functions. Another objective to use topological map is to map high-dimensional input space onto low-dimensional output map. Each map node (or neuron) consists of similar data. The topological map is a simple data representation which gives advantage in data visualization. Other SOM variants propose to represent topological map as a hierarchical tree. Those methods are different from each other in many aspects. Indeed, the tree-like structure is an efficient and optimal representation facilitating cluster analysis.

Many hierarchical SOM variants for visualization have been developed up to date. In (Samsonova et al., 2006) TreeSOM is employed to deal with protein data. TreeSOM proposes a clustering as a consensus tree that represents reliable clusters as subtrees. Using tree representation, a cluster is defined as a group of nodes surrounded by an uninterrupted border of a given shade or darker representing distances equal to or greater than a given distance threshold. At each threshold during learning one or more clusters are split into several sub-clusters that is represented as a tree node. AutoSOME (Newman et Cooper, 2010) used SOM to achieve both a dimensional reduction in gene expression modules and an initial organization of the input data. AutoSOME provides data visualization using network diagrams and differential heat maps. H^2SOM (Ontrup et Ritter, 2005) is based on a growing scheme training an unique lattice structure that has a hyperbolic form. Each node in the periphery of the existing network can be expanded to build large size structure. SOM-AT (Peura, 1999) is based on introducing a distance metric and adjusting schemes for attribute trees. The most optimal tree is selected to represent input data. Evolving Tree (Pakkanen) is another SOM variant whose nodes are arranged in a tree topology that grows over time. This growing structure uses the shortest path between two nodes in a tree as the neighborhood function for the self-organizing process.

In this paper we present clustering and visualization technique called SoT that provides a hierarchical and topological structure. Our method seeks to build multiple hierarchical trees without losing data topology. The proposed structure consists of a topological map and a fixed number of hierarchical trees. Each map node is represented by a prototype associated with a hierarchical tree. It can be exploited by descending from topological level (map) to the last level of trees, which provides relevant information about the clustering and data topology.

2 Definitions and notations

2.1 Graph and Laplacian matrix

We denote $G(V, E)$ an undirected graph, a set of N vertices V of and a set of edges E. A direct link between two vertices v_i and $v_j \in V$ creates an edge $\Leftrightarrow (v_i, v_j) \in E$. Let W be the adjacency matrix of G :

$$W(i,j) = \begin{cases} 1 & \text{if } (v_i, v_j) \in E, \forall i, j = 1, .., N \text{ and } i \neq j \\ 0 & \text{otherwise.} \end{cases}$$

The adjacency matrix contains only binary information of the connectedness between vertices of G. Given $deg(i)$ the degree of the vertex v_i measuring the number of incident edges to v_i. The degree matrix D is a diagonal matrix defined as :

$$D(i,j) = \begin{cases} deg(i) & \text{if } i = j, \forall i, j = 1, .., N \\ s0 & \text{otherwise.} \end{cases}$$

These matrices are not considered as a metric that explicitly calculate either an appropriate similarity or distance measure. Thus, the Laplacian matrix (Chung, 1997; Luxburg, 2007) is used. The Laplacian matrix denoted by L is a representation of graph through the adjacency and degree matrix.

$$L = D - W \tag{1}$$

The study of the Laplacian called spectral graph theory has pointed out that some of the underlying structure can be seen in the properties of the Laplacian. An eigenvector or a combination of several eigenvectors of L is enough to compute vertex similarity or geometric distance.

Following spectral graph theory (Chung, 1997; Luxburg, 2007), we take the first d smallest eigenvectors $A = e_1, .., e_d$ ($\forall i = 1..d, e_i \in \mathbf{R}^N$) of the Laplacian matrix. In this case, a line (or a vector) $\mathbf{x}_i \in A$ respectively illustrates a vertex $v_i \in V$ in the new space. Thus we have reduced and mapped a graph G into A where each vertex of G is represented by a respective vector of dimension d and the similarity between two vertices $sim(v_i, v_j)$ can be defined as the distance (Euclidean distance) between their respective vectors $\mathbf{x}_i, \mathbf{x}_j$.

2.2 Definitions

We want to define some necessary definitions in our paper.

— A vertex $v_i \in G$ is represented by a vector $\mathbf{x}_i \in A$. This vector is represented by a tree node in the SoT structure.

— A map node (or cell c) in grid is associated with its respective prototype. In the grid, prototypes are randomly initialized.

— Each cell c in the grid is associated with its own tree denoted $tree_c$.

— The root node (or support) is an *empty* node to whom the first tree nodes are going to connect.

3 Self-organizing Trees model

3.1 Self-organizing trees and clustering discovery with hierarchical relations

Our idea is to generate a set of trees arranged according to certain topology. The map preserves the data topology while data similarity lies on the hierarchical structure. The map nodes represent the prototypes in the grid. Unlike other hierarchical representation where only the leaf nodes contain data information, every tree node in SoT structure corresponds to input

data.

The proposed algorithm is able to detect clusters and represent these clusters as topological and hierarchical structure. Here each tree is considered as a cluster and a sub-tree from any branch as sub-cluster. Confidence of each cluster may be easily observed because of hierarchical relations between data.

3.2 Principle : Topological and hierarchical clustering

Our model SoT seeks to find an automatic clustering that provides hierarchical and topological organizations of data. The main idea is to reconstruct dataset in a 2D space by representing data in hierarchical way as several auto-organized trees. The basic principle of tree construction is inspired from the self-assembly rules of AntTree method (Azzag et al., 2007). Connection of a tree node to another one in tree structure is based on their similarity (i.e. Euclidean distance). The rules will be detailed in Section 3.3.

The size of the grid $\mathcal{W}$ denoted by $k = m \times n$ must be provided a priori. So we have $\mathbf{x}_i = (x_i^1, x_i^2, ..., x_i^d) \in \mathcal{R}^d$ where $i = [1, .., N]$. A variety of self-organizing models is derived from the first original model proposed by Kohonen (Kohonen et al., 2001). All models are different from each other but share the same idea : depict large data-sets on a simple geometric relationship projected on a reduced topology (1D or 2D). The proposed model uses the same grid process, combined with a new concept of neighborhood and hierarchical tree construction. This grid has topological order of $k\ trees$.

Self-organizing process requires neighborhood functions to preserve topological relationships between trees. Hence the neighborhood functions are needed to update trees. For each pair of $tree_c$ and $tree_r$ on the map, their mutual influence is defined by the function $\mathcal{K}^T(\delta(c,r)) = \exp(\frac{-\delta(c,r)}{T})$ where T represents the temperature function that controls the size of the neighborhood and $\delta(c,r)$ is defined as the shortest distance between r and c on the grid $\mathcal{W}$. We associate to each $tree_c$ a prototype denoted $prototype(c)$. The cost function of self-organizing tree is expressed as :

$$\mathcal{R}(\phi, \mathcal{L}) = \sum_{c=1}^{k} \sum_{i \in tree_c} \sum_{r=1}^{k} \mathcal{K}^T(\delta(\phi(subtree_{\mathbf{x}_i}), r)) ||\mathbf{x}_i - \mathbf{x}_{prototype(r)}||^2 \quad (2)$$

where $\mathcal{L} = \cup_{r=1}^{k} prototype(r)$, ϕ is the assignment function and $subtree_{\mathbf{x}_i}$ contains $\mathbf{x}_i$ and all tree nodes recursively connected to it ; $\forall \mathbf{x}_j \in subtree_{\mathbf{x}_i}$, $\phi(\mathbf{x}_j) = \phi(\mathbf{x}_i)$.

$$\phi(subtree_{\mathbf{x}_i}) = \arg\min_{r} \sum_{c=1..k} \mathcal{K}^T(\delta(r,c)) \left\|\mathbf{x}_i - \mathbf{x}_{prototype(c)}\right\|^2 \quad (3)$$

Minimizing cost function $\mathcal{R}(\phi, \mathcal{L})$ is a combinatorial optimization problem. In this work we propose to minimize the cost function in the same way as "batch" version using statistical characteristics provided by trees to accelerate the convergence of the algorithm. These characteristics are used in assignment function 3.

3.3 SoT batch algorithm

Firstly we define some notations necessary for the algorithms. A tree node has only one among three status at a time :

1. *Initial* : the default status before training ;
2. *Connected* : the tree node is currently connected to another node ;
3. *Disconnected* : the tree node were connected at least once but now gets disconnected.

Let us denote $\mathbf{x}_{pos}$ is the support (the root of tree) or the tree node position where $\mathbf{x}_i$ is located. At the beginning, $\mathbf{x}_i$ is located on the support and will move in the tree (toward another tree nodes) in order to find the best position ; $\mathbf{x}^+$ and $\mathbf{x}^-$ two tree nodes connected to $\mathbf{x}_{pos}$ which are respectively the most similar and dissimilar tree node to $\mathbf{x}_i$. Let $list$ denote a list of tree nodes. Before training, $list$ contains only *initial* nodes, whenever a tree node becomes *connected*, it is immediately removed from $list$; alternatively whenever a tree node and its children get disconnected from a tree, we put them back onto $list$. After several iterations, $list$ might contain both *initial* and *disconnected* tree nodes.

The summary of batch version is given as in Algorithm 1. Our batch algorithm can proceed by alternating between three steps : assignment, tree construction and update.

Assignment step

There are two distinct types of assignment :

1. For one single tree node $\mathbf{x}_i$, it can be assigned using $\phi(\mathbf{x}_i)$ (see Line 6 in Algorithm 1).
2. In the case of a $subtree_{\mathbf{x}_i}$ ($subtree_{\mathbf{x}_i}$ consists of the sub-tree with $\mathbf{x}_i$ as the root) (see Line 16 in Algorithm 1).

An example where multiple tree nodes are simultaneously assigned is shown in Fig. 1. Due to the last update, $subtree_{\mathbf{x}}$ consists of three violet tree nodes that are no longer connected to $tree_{c_{old}}$. Now we have to determine the new best match $tree_c$ for the tree node $\mathbf{x}$ using $\phi(subtree_{\mathbf{x}})$. The assignments of child nodes of $\mathbf{x}$ follow automatically the one of $\mathbf{x}$ using the statistical characteristics of tree (each data is connected to 1-NN). Even though these three nodes are now found in the new cell c, their hierarchy in new cell remains in the old cell c_{old}.

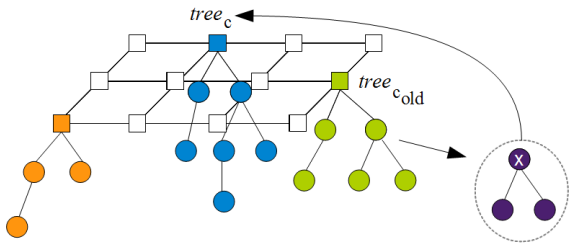


FIG. 1 – *Group assignment example from* $tree_{c_{old}}$ *to* $tree_c$

Tree construction step

This step is realized by the function *constructTree* in Algorithm 2. This states the rules for connecting and disconnecting tree nodes. Let us denote by $T_{Dissim}(\mathbf{x}_{pos})$ the lowest similarity

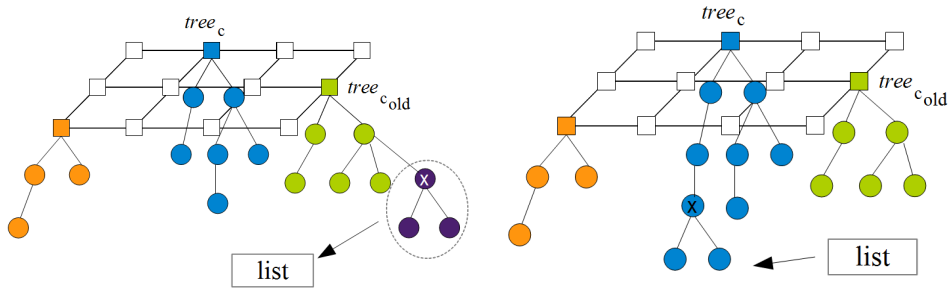
Algorithm 1 SoT batch algorithm

```
1: initialize k prototypes
2: while the stopping criteria are not yet fulfilled do
3:    initialize list
4:    while list is not empty do
5:       x_i is the first data from list
6:       c ← φ(x_i)
7:       if x_i is initial then
8:          tree_c ← constructTree(tree_c, x_i)
9:       end if
10:      if x_i is disconnected then
11:         tree_c ← constructTree(tree_c, x_j),
12:         ∀x_j ∈ subtree_{x_i}
13:      end if
14:      if x_i is connected and c ≠ c_old then
15:         subtree_{x_i} ← disconnect x_i and all tree nodes recursively connected to x_i ;
16:         tree_c ← constructTree(tree_c, x_j), ∀x_j ∈ subtree_{x_i}
17:      end if
18:      if x_i is not connected then
19:         put x_i at the end of list ;
20:      else
21:         remove subtree_{x_i} from list.
22:      end if
23:      update prototypes
24:   end while
25: end while
```

value, which can be observed among the children of $\mathbf{x}_{pos}$. $\mathbf{x}_i$ is connected to $\mathbf{x}_{pos}$ if and only if the connection of $\mathbf{x}_i$ decreases further this value. Since this minimum value can only be computed with at least two tree nodes, then the first two tree nodes are automatically connected without any test (Line 1 in Algorithm 2). This may result in "abusive" connections for the second tree node. Therefore the second is removed and disconnected as soon as the third one is connected (Line 4). For this latter node, we are certain that the similarity test has been successful. Once this second one has been removed, all nodes connected to it are dropped, and placed back onto *list*. They may be assigned to other trees later using the same rules. For each $\mathbf{x}_{pos}$, we allow to disconnect only once to assume the convergence of the algorithm. If we have already disconnected data from $\mathbf{x}_{pos}$, the Line 13 is employed. During the training, two cases of disconnection must be distinguished :

1. disconnect tree node(s) due to the assignment (Line 15 in Algorithm 1),
2. disconnect tree node when a node comes into play (Line 7 in Algorithm 2).

All nodes disconnected from tree are placed back onto *list* and they will reconnect to another tree or stay in the same tree. A simple example of disconnection/reconnection for a group of nodes (or sub-tree) is illustrated in Fig. 2. Given $tree_{c_{old}}$ as in Fig. 2a, the tree node $\mathbf{x}$ is to disconnect from this tree. All the nodes connected to $\mathbf{x}$ must be recursively disconnected too ;



(a) Disconnect nodes from $tree_{c_{old}}$

(b) Reconnect nodes to $tree_c$

FIG. 2 – *Example of disconnecting/reconnecting a sub-tree.*

in this case it should be two child nodes of $\mathbf{x}$. Therefore $subtree_{\mathbf{x}}$ (three violet nodes) have *disconnected* status and are put back onto the pool *list*. After getting new assignment, $\mathbf{x}$ is going to connect to $tree_c$. It leads to that the child node of $\mathbf{x}$ have $tree_c$ as their best match tree too. We systematically connect this sub-tree to $tree_c$ and the result is shown in Fig. 2b.

Update step

This step is necessary to compute the new prototypes. We can easily adapt our algorithm to choose a representative prototype. Prototype depends on data structure, it can be seen as centroid in case of traditional continuous datasets or as leaders (Stanoev et al., 2011) in case of graph datasets. The prototype expression is as following

$$prototype_c = \arg \max_{\forall i \in tree_c} (deg(v_i)) \tag{4}$$

where the local degree $deg(v_i)$ is the number of internal edges incident to v_i. Choosing a representative prototype allows easily adapting our algorithm.

3.4 Computational complexity

Algorithm 1 requires k iterations to reach stopping criteria. During one iteration, an operation includes three iterative steps : assignment, tree construction and updating support. The number of assignments starts from 100% and decreases taking advantage from the tree structure (the subtree nodes follow the node root assignment), which allows the run time to be reduced (See Assignment step section). To terminate training for N objects, these steps require $N \log N$ operations. Finally, SoT has a complexity of $\theta(kN \log N)$ which is competitive with traditional topological map.

Algorithm 2 constructTree($tree$,$\mathbf{x}_i$)

if 1 less than 2 data connected to $\mathbf{x}_{pos}$ **then**
  connect $\mathbf{x}_i$ to $\mathbf{x}_{pos}$
end if
if 2 more than 2 data connected to $\mathbf{x}_{pos}$ and for the first time **then**
  $T_{Dissim}(\mathbf{x}_{pos}) = min(sim(\mathbf{x}_i, \mathbf{x}_j))$
  {w}here $\mathbf{x}_i$ and $\mathbf{x}_j$ are any pair of data connected to $\mathbf{x}_{pos}$
  { $sim(\mathbf{x}_i, \mathbf{x}_j) = ||\mathbf{x}_i - \mathbf{x}_j||^2$, $\mathbf{x}_i, \mathbf{x}_j$ are normalized}
  if $sim(\mathbf{x}_i, \mathbf{x}^+) < T_{Dissim}(\mathbf{x}_{pos})$ **then**
    disconnect $subtree_{\mathbf{x}^-}$ from $\mathbf{x}_{pos}$
    {disconnect recursively all childs of $\mathbf{x}^-$}
    connect $\mathbf{x}_i$ to $\mathbf{x}_{pos}$
  else
    move $\mathbf{x}_i$ to $\mathbf{x}^+$
  end if
end if
if 3 more than 2 data connected to $\mathbf{x}_{pos}$ and for the sencond time **then**
  if $sim(\mathbf{x}_i, \mathbf{x}^+) < T_{Dissim}(\mathbf{x}_{pos})$ **then**
    connect $\mathbf{x}_i$ to $\mathbf{x}_{pos}$
  else
    move $\mathbf{x}_i$ to $\mathbf{x}^+$
  end if
end if
return $tree$.

4 Experimentation on Protein Dataset

4.1 Protein domains

Proteins are large polymeric molecules made of one or more amino acids chains. They support a wide range of biological functions such as structure, transport, binding, catalysis, and so on. The function of a protein is closely related to the three dimensional (3D) arrangment (fold) of its chain into a compact structure. This structure can be described at many levels : locally at the atomic level to understand biochemical processes and more globally at the domain level to understand evolutionary processes. A protein domain can be define as a compact module of the protein structure shared by multiple proteins. Usually a protein can be made by one or more domains where each domain may support a different role of the protein function. Protein domains are considered to have a specific sequence and structure evolution pattern.

4.2 Expert classification of protein domains

The protein domain data set used in this study derives from the SCOP_v_1.75 classification. It contains 110800 domains described in term of sequences relying to 38221 solved 3D structures of the Protein Data Bank. They are organized in a 4-level hierarchical classification, namely the "Class", "Fold", "Super Family" and "Family" levels, organizing the domains

by increasing similarities. For example, domains in the same "Class" only share very coarse structural similarities (secondary structure composition) and domains of the same "Family" are selected to share a high similarity level in term of length and fine spatial organization of the amino-acids.

4.3 Protein structure similarities

From these 110800 protein domains we kept the ASTRAL_40 subset (Brenner et al., 2000) : the 3D structures of SCOP domains presenting less than 40% sequence identity. For each pair of protein domains, a structure alignment is done with Yakusa (Carpentier et al., 2005). This algorithm seeks to find the set of longest common sub-structures between a query protein domain and any protein of a data base. Each alignment is evaluated in term of a z-score measuring the significance of spatially compatible aligned blocks relatively to found alignments : higher is the z-score, more significant is the alignment and more similar the protein domains are considered.

4.4 Graph of pairwise similarities

Let $G_z = (V, E)$ be the graph of pairwise protein domain structures similarities defines as in (Santini et al., 2012). Each vertex $v_i \in V$ represents a protein domain, and an edge (v_i, v_j) occurs between two domains if their alignment provided by Yakusa presents a $z\_score > z$. With : $z\_score = \frac{s_i - \overline{S}}{\sigma_S}$. Fig. 3 displays the original graph Astral G_7 and G_6. The protein domains from the same SCOP "Class" in the 4-level hierarchical classification have the same tone of color, i.e all the "alpha" proteins are associated with red tone ; all the "beta" proteins are associated with green tone ; all the "alpha and beta" proteins are associated with blue tone ; etc.... Within each tone, each SCOP "Family" has a unique color.

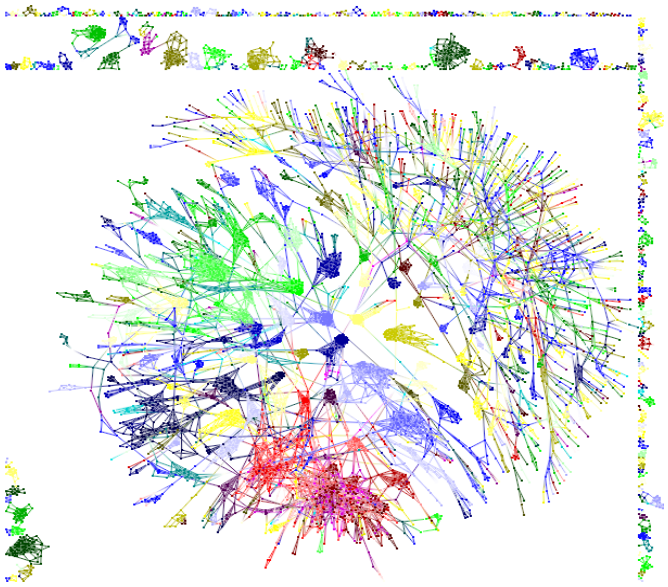
It is clear that both graphs show the difficulties to explore the graph particularly with the graph G_6. Thus, this is one of the motivations to use SoT algorithm, which allows decomposing the original graph into another clear structure. This structure has the distinction of being topological and hierarchical.

5 Discussions on experimental results

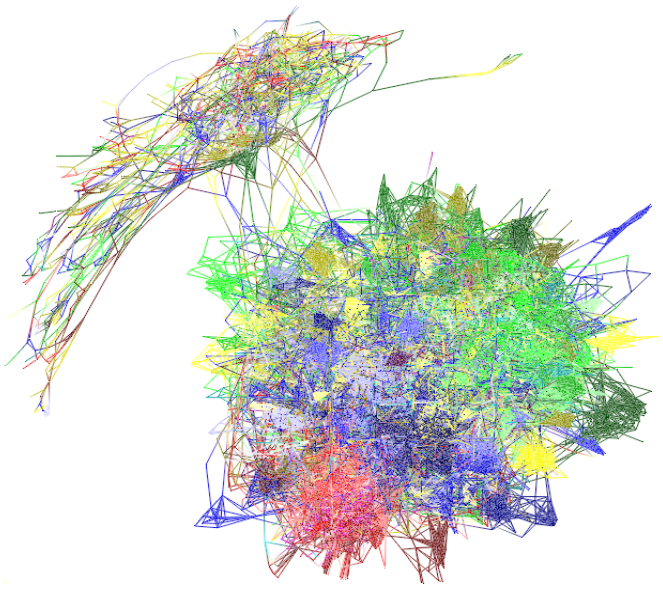
5.1 Study on the removed and synthetized edges

In our experiments , we fix the number of smallest eigenvectors given by the Laplacian $d = k$ (k is the number of trees or prototypes). Given two vertices v_i and v_j with four labels according to the 4-level hierarchical SCOP classification $C(v_i) = \{l, m, n, o\}$ and $C(v_j) = \{l', m', n', o'\}$ where l, m, n and o (ie l', m', n' and o') stand respectively for "Class", "Fold", "Super Family" and "Family" SCOP identificators, we define the distance $dist$ between two vertices as a function of their SCOP classification as following :

— $dist(v_i, v_j) = 0$ if $C(\mathbf{x}_i) = C(\mathbf{x}_i)$ or $l = l'; m = m'; n = n'; o = o'$
— $dist(v_i, v_j) = 1$ if $l = l'; m = m'; n = n'; o \neq o'$
— $dist(v_i, v_j) = 2$ if $l = l'; m = m'; n \neq n'; o \neq o'$
— $dist(v_i, v_j) = 3$ if $l = l'; m \neq m'; n \neq n'; o \neq o'$



(a) G_7



(b) G_6

FIG. 3 – *Original graph of the Astral G_7 and G_6 dataset. Each SCOP "Family" has a color in the SCOP "Class" color tone.*

TAB. 1 – *The ratio of removed edges R^{dist}, $dist$ is the distance between two proteins in 4-level classification.*

R^{dist} (n_{obs})	dist	4	3	2	1	0	N_{obs}
G_7	k = 900	1.0652 (653)	1.0786 (490)	1.0590 (287)	1.0587 (3274)	0.9763 (11257)	15961
	k = 2000	0.9880 (629)	1.0184 (477)	1.0396 (292)	1.0483 (3281)	0.9838 (11489)	16165
	n_{origin}	699	518	309	3526	13147	18199
G_6	k = 900	1.0325 (2789)	1.0335 (1999)	1.0215 (738)	1.0136 (6558)	0.9855 (17402)	29486
	k = 2000	1.0194 (2793)	1.0179 (1997)	1.0180 (746)	1.0087 (6620)	0.9911 (17752)	29908
	n_{origin}	2793	2000	747	6690	18258	30488

— $dist(v_i, v_j) = 4$ if $l \neq l'; m \neq m'; n \neq n'; o \neq o'$

The ratio $R^{dist} = \frac{n_{obs}^{dist}/N_{obs}}{n_{origin}^{dist}/N_{origin}}$ is a measure of how SoT removes edges in function of distance $dist$ during learning, where N_{origin} is the total number of edges in graph G_z, N_{obs} the total number of edges removed from G_z by SoT algorithm, n_{origin}^{dist} the number of G_z edges between vertices at distance $dist$ and n_{obs}^{dist} the number edges removed from G_z between vertices at distance $dist$. As shown in Table 1 for different values of k and z, we observe that the smallest value of R is reached when $dist = 0$. So for each value of k, SoT removes relatively less edges between data within the SCOP "Family".

TAB. 2 – *Number of synthesized edges with respect with the distance for each value of k in the G_7 and G_6 graph.*

	dist	4	3	2	1	0	Total
G_7	k = 900	950	477	92	509	611	2639
	k = 2000	633	189	22	80	183	1107
G_6	k = 900	3307	1453	130	365	394	5649
	k = 2000	3049	1314	69	235	265	4932

As was explained in Section 3.3, SoT creates a tree topology structure where each tree node represents a data. Deleting links is not enough to reach this structure, thus SoT also creates synthesized edges to build tree structure. In the original graph, these synthesized edges allow to find a path between the tree nodes belonging to the same class. Table 2 lists the number of synthesized edges for different values of k. It is difficult to see the result on the original graph, but it is clear that the tree structure provided by SoT in Fig. 5a and 7a decreases complexity of visualization and exploration of these graphs. Here we provide various views for data visualization which proofs that our method can offer more useful information. Tulip (Auber, 2003) is used as the framework for the visualization. In this section, we only use $k = 900$ to visualize the Astral G_7 and G_6 datasets.

Fig. 4 and 6 present tree topology view of Astral G_7 and G_6 respectively. The map cells (squares) are located in the center surrounded by trees. Figure 5 and 7 show two zooms of a region in the SoT structure and a related zone of the original graph involving same data points for comparison. It is easy to note that the Fig. 5a and 7a allow analyzing data better than the

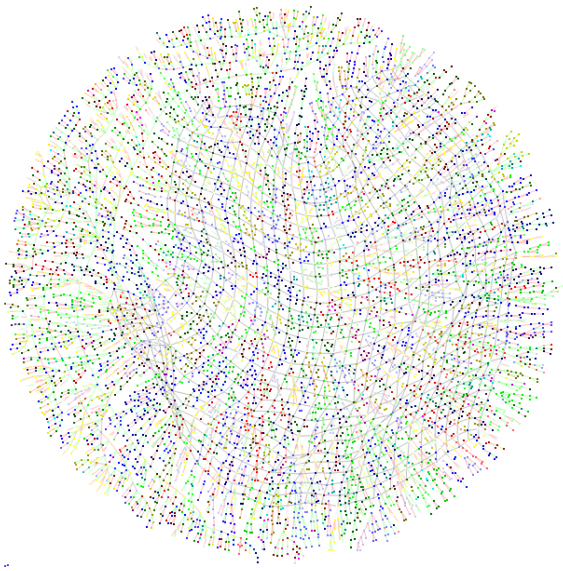
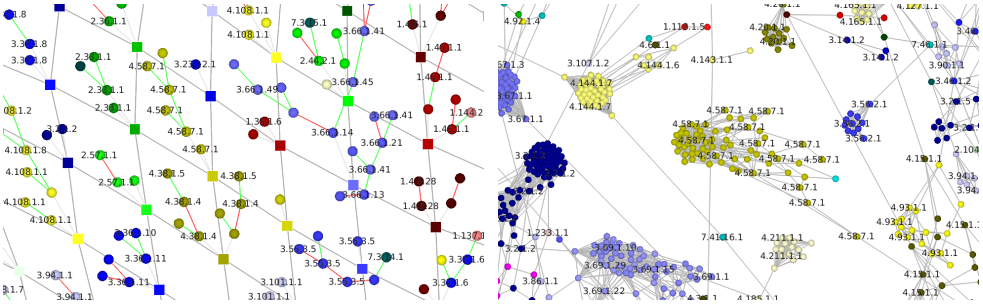


FIG. 4 – *Astral G_7 - SoT : hierarchical and topological structure. Each node has respectively its color in the original graph corresponding to SCOP class.*



(a) SoT structure extracted from Fig. 4 (b) Original graph extracted from Fig. 3a

FIG. 5 – *Sample visualizations for $z = 7$ centered on cluster 4.58.7.1*

traditional visualizations presented in Fig. 5b and 7b.

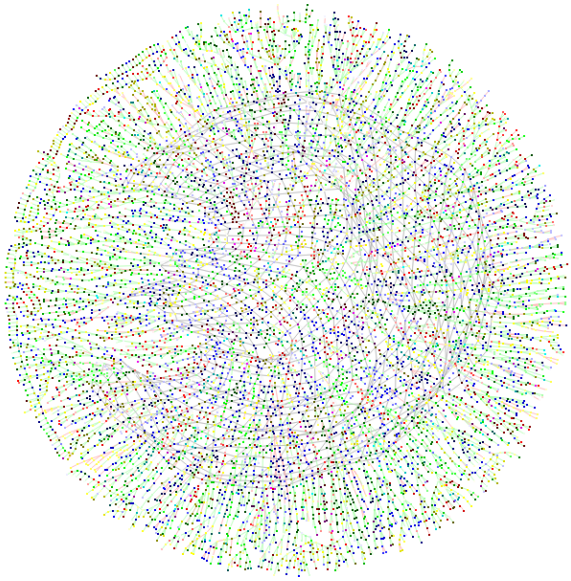
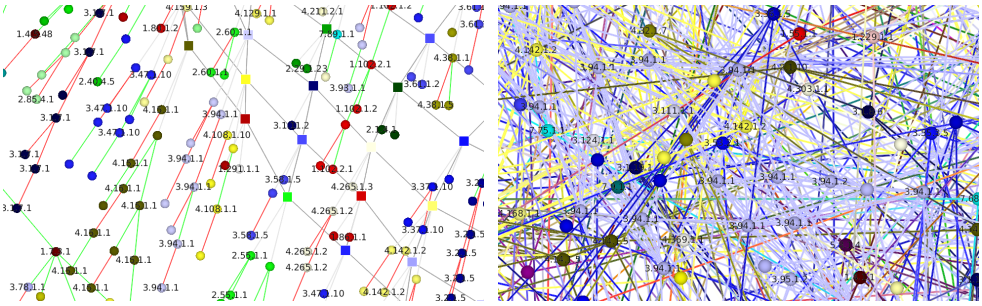


FIG. 6 – *Astral G_6 - SoT : hierarchical and topological structure. Each node has respectively its color in the original graph.*



(a) SoT extracted from Fig. 6 (b) Original graph extracted from Fig. 3b

FIG. 7 – *Sample visualizations for $z = 6$ centered on cluster 3.91.1.1.*

5.2 Clustering quality

To evaluate the clustering quality, NMI (Normalized Mutual Information) and Rand index are computed and presented in Table 4.

Given a set of N objects of l classes classified into k clusters and two partitions to com-

pare : $X = \{x_1, .., x_N\}$ where $x_i \in [C_1..C_k]$ a random variable for cluster assignments, $Y = \{y_1, .., y_N\}$ where $y_j = [B_1..B_l]$ a random variable for the pre-existing labels. The contingency table can be expressed as in Table 3.

TAB. 3 – *Contingency table*

B\C	C_1	C_2	$\cdots$	C_i	$\cdots$	C_k	Sum
B_1	n_{11}	n_{12}	$\cdots$	n_{1i}	$\cdots$	n_{1k}	N_{1*}
B_2	n_{21}	n_{22}	$\cdots$	n_{2i}	$\cdots$	n_{2k}	N_{2*}
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
B_j	n_{j1}	n_{j2}	$\cdots$	n_{ji}	$\cdots$	n_{jk}	N_{l*}
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
B_l	n_{l1}	n_{l2}	$\cdots$	n_{lk}	$\cdots$	n_{lk}	N_{l*}
Sum	N_{*1}	N_{*2}		N_{*i}		N_{*k}	N

The Normalized Mutual Information is given by the following expression :

$$NMI = \frac{\sum_j \sum_i n_{ji} log_2(\frac{N.n_{ji}}{N_{j*}N_{*i}})}{(\sum_j N_{j*} log_2(\frac{N_{j*}}{N}))(\sum_i N_{*i} log_2(\frac{N_{*i}}{N})}) \tag{5}$$

The Rand Index is defined straightforwardly as

$$Rand = (N_{00} + N_{11}) / \left(\begin{array}{c} N \\ 2 \end{array} \right) \tag{6}$$

where N_{11} is the number of pairs that are in the same cluster in both B and C; N_{00} is the number of pairs that are in different clusters in both B and C.

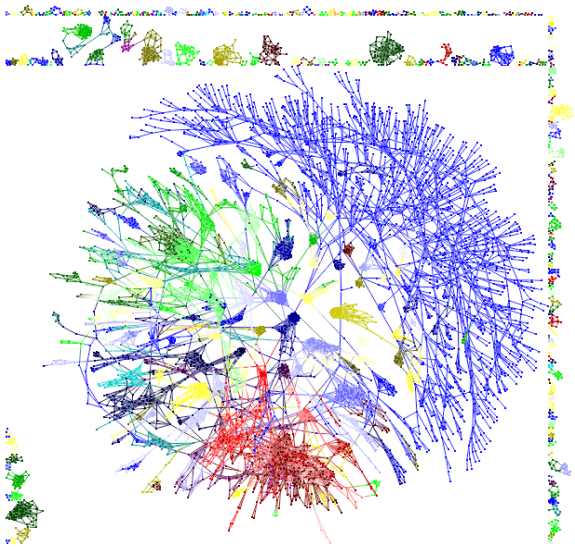
SoT provides a good clustering : the Rand values are close to 1. Moreover we can compare the clustering quality provided by SoT in visual aspect. Fig. 8 shows the original graph of the Astral G_7 and G_6 dataset after applying the majority vote rule. These figures enable the difference in color with Fig. 3.

TAB. 4 – *Quality measure (NMI and Rand) for G_7 and G_6*

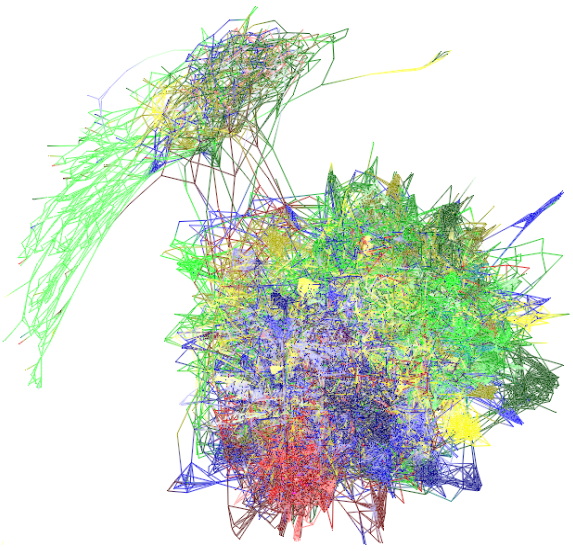
	# Node	k	NMI	Rand
G_7	8227	900	0.943	0.781
		2000	0.941	0.805
G_6	6606	900	0.994	0.731
		2000	0.994	0.739

6 Conclusion

Bioinformatics requires handling large volumes of data, involving natural interaction with information science. Difficulties in dealing with this dataset motived us to introduce an enhanced topological and hierarchical structure visualization, in order to make easier to deal



(a) G_7



(b) G_6

FIG. 8 – *Original graph of the Astral G_7 and G_6 dataset. Node color is determined by a color using majority rules.*

with protein dataset by taking advantage of the characteristics of topological map. Pairwise structural similarities between protein domains are by nature local, potentially multiple, not compatible and non transitive. As a consequence, depicting these complex and noisy relations in a intelligible way is challenging as shown by the graph of similarities.

In this context, SoT approach revealed to be very efficient. It allows the exploration of a large amount of real data. It simplifies the representation of structural neighborhoods, not only by reducing and reorganizing the links between protein domains, but also by providing a relevant clustering of 3D structures coherent to the stand of the art expert classifications. In addition, tree organization that is provided by SoT algorithm is of great interest and should be investigated in a further analysis. In particular the hierarchy of structure and the derived distances that could be computed between proteins from these trees could give evolutionary informations helping to investigate processes that drive the conservation or the divergence of protein structure and function.

The idea of the self-organizing tree for protein dataset proposed in this article certainly needs more investigations and testing. We want to perform a user-study to highlight the possible application of the proposed method to interactive clustering. The main advantage of Interactive Protein Data Clustering is that the expert has a high confidence in these results because he/she has visually validated them.

Références

Andreeva, A., D. Howorth, J. Chandonia, S. Brenner, T. Hubbard, C. Chothia, et A. Murzin (2008). Data growth and its impact on the scop database : new developments. *Nucleic Acids Res 36*(Database issue), D419–25.

Auber, D. (2003). Tulip : A huge graph visualisation framework. In P. Mutzel et M. Jünger (Eds.), *Graph Drawing Softwares*, Mathematics and Visualization, pp. 105–126. Springer-Verlag.

Azzag, H., G. Venturini, A. Oliver, et C. Guinot (2007). A hierarchical ant based clustering algorithm and its use in three real-world applications. *European Journal of Operational Research 179*(3), 906–922.

Bernstein, F., T. Koetzle, G. Williams, E. Meyer, M. Brice, J. Rodgers, O. Kennard, T. Shimanouchi, et M. Tasumi (1977). The protein data bank. a computer-based archival file for macromolecular structures. *Eur. J. Biochem. 80*(2), 319–24.

Brenner, S. E., P. Koehl, et M. Levitt (2000). The astral compendium for protein structure and sequence analysis. *Nucleic Acids Research 28*(1), 254–256.

Carpentier, M., S. Brouillet, et J. Pothier (2005). Yakusa : a fast structural database scanning method. *Proteins 61*(1), 137–51.

Chung, F. R. K. (1997). *Spectral Graph Theory (CBMS Regional Conference Series in Mathematics, No. 92)*. American Mathematical Society.

Doan, N.-Q., H. Azzag, et M. Lebbah (2012). Self-organizing map and tree topology for graph summarization. In *ICANN (2)*, pp. 363–370.

Kohonen, T. (2001). *Self-Organizing Maps*. Berlin : Springer.

Kohonen, T., M. R. Schroeder, et T. S. Huang (Eds.) (2001). *Self-Organizing Maps* (3rd ed.). Secaucus, NJ, USA : Springer-Verlag New York, Inc.

Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and Computing 17*, 395–416.

Murzin, A., S. Brenner, T. Hubbard, et C. Chothia (1995). Scop : a structural classification of proteins database for the investigation of sequences and structures. *J Mol Biol 247*(4), 536–40.

Newman, A. M. et J. B. Cooper (2010). Autosome : a clustering method for identifying gene expression modules without prior knowledge of cluster number. *BMC Bioinformatics 11*, 117.

Ontrup, J. et H. Ritter (2005). A hierarchically growing hyperbolic self-organizing map for rapid structuring of large data sets. In *Proceedings of the 5th Workshop on Self-Organizing Maps (WSOM 05)*, Paris, France.

Pakkanen, J. The evolving tree : a new kind of self-organizing neural network.

Pearl, F. M. G., C. F. Bennett, J. E. Bray, A. P. Harrison, N. Martin, A. Shepherd, I. Sillitoe, J. Thornton, et C. A. Orengo (2003). The cath database : an extended protein family resource for structural and functional genomics. *Nucl. Acids Res. 31*(1), 452–455.

Peura, M. (1999). The self-organizing map of attribute trees.

Samsonova, E. V., J. N. Kok, et A. P. Ijzerman (2006). Treesom : Cluster analysis in the self-organizing map. neural networks. *American Economic Review 82*, 1162–1176.

Santini, G., H. Soldano, et J. Pothier (2012). Automatic classification of protein structures relying on similarities between alignments. *BMC Bioinformatics 13*, 233.

Stanoev, A., D. Smilkov, et L. Kocarev (2011). Identifying communities by influence dynamics in social networks. *CoRR abs/1104.5247*.

Summary

Clustering and visualizing multi-dimensional or structured data are important tasks for data analysis, especially in bioinformatics. Self-organizing models are often used to address both of these issues. In this paper we introduce a hierarchical and topological visualization technique called Self-organizing Trees (SoT) which is able to represent data in hierarchical and topological structure. The experiment is conducted on a real-world protein data set.

Construction et exploration d'un graphe de proximité portant sur une grande collection de jeux de données publiques

Tianyang Liu*, Fatma Bouali**,*, Gilles Venturini*

* Université François-Rabelais de Tours, Laboratoire d'Informatique
64 avenue Jean Portalis, 37200 Tours, France,
tianyang.liu@etu.univ-tours.fr, venturini@univ-tours.fr
** Université de Lille2, IUT, Dpt STID
25-27 Rue du Maréchal Foch, 59100 Roubaix, France, fatma.bouali@univ-lille2.fr

Résumé. Nous résumons dans cet article et cette présentation nos travaux sur la construction d'une carte visuelle et interactive d'une grande collection de jeux de données ouvertes (article de revue internationale en cours de publication). Notre approche s'appuie sur 1) la représentation vectorielle de ces jeux de données en utilisant des méthodes de fouille de textes, 2) la construction d'un graphe de proximité en utilisant une approche parallèle, 3) la visualisation interactive du graphe obtenu dans le logiciel Tulip. Nous avons réalisé de nombreuses expériences pour étudier les limites et propriétés de notre approche.

1 Introduction

De nombreux pays ont récemment rendu leurs données publiques. En général, de très grands volumes d'information sont publiés en ligne par les gouvernements et d'autres fournisseurs de données. Dans le cas des données ouvertes françaises, plus de 350 000 jeux de données ouvertes (JDO) sont accessibles au public. En général, les sites web des gouvernements offrent aux utilisateurs des mécanismes de recherche limités, reposant sur des moteurs de recherche traditionnels. Des possibilités de visualisation ou d'interaction sont rarement offertes aux utilisateurs de ces sites. Avec, un tel accès à l'information, et compte tenu du grand nombre de JDO disponibles, les utilisateurs peuvent avoir des difficultés à trouver des jeux de données pertinents et correspondant à leurs objectifs.

Le contexte de cette étude porte donc sur l'aide pouvant être apportée aux utilisateurs pour mieux explorer une grande collection de JDO. En explorant un ensemble de données, les utilisateurs doivent obtenir une vue d'ensemble d'abord, puis des détails à la demande : "overview first, ..., then details on demand" (Shneiderman, 1996). Par conséquent, notre premier objectif est de fournir un aperçu d'un grand ensemble de JDO. Cette vue d'ensemble devrait être visuelle et interactive. Elle devrait révéler des informations globales, telles que la présence de clusters, les relations entre ces clusters, leurs tailles, les sujets traités, les informations atypiques, etc. Une telle vue d'ensemble des données peut être utile pour les utilisateurs souhaitant

analyser globalement la collection de données et trouver des JDO portant sur les thèmes recherchés. Un deuxième objectif de notre travail est de fournir des détails sur des clusters et des jeux de données et d'effectuer des suggestions aux utilisateurs pour explorer localement et de façon détaillée des jeux de données. Lorsqu'un utilisateur consulte un jeu de données, il peut être intéressé par explorer d'autres jeux ayant des contenus similaires.

2 Etat de l'art et résumé de l'approche proposée

La fouille de données ouvertes étant un domaine récent, il n'y a pas encore d'approches visuelles et interactives spécifiques qui pourraient être directement utilisées pour atteindre nos objectifs. Cependant, plusieurs techniques issues de différents domaines peuvent être pertinentes pour le problème abordé. Nous devons d'abord extraire les informations et les caractéristiques des JDO afin de les représenter dans un espace de caractéristiques. Par ailleurs, nous devons trouver comment construire une vue d'ensemble de ces JDO ainsi que les liens existants entre eux selon leurs similarités. Nous devons aussi utiliser une interface utilisateur qui peut représenter une carte globale des données et offrir aux utilisateurs des possibilités d'interactions. Cette interface doit également fournir des informations détaillées sur un JDO particulier. Enfin, la taille de la collection (plusieurs centaines de milliers de JDO) aura un impact important. Par conséquent, nous nous sommes inspirés de plusieurs approches dans le traitement des données, de la fouille de texte, l'apprentissage topologique, la programmation parallèle sur une architecture multi-coeurs et la visualisation de graphes pour produire notre outil.

Notre approche peut être résumée comme suit :

1. Les JDOs sont téléchargés à partir du site Web et pré-traités : leurs méta-données et leurs contenus (fichier de type tableur) sont extraits et re-codés dans un format XML,
2. Des caractéristiques textuelles sont extraites des méta-données et du contenu. Elles sont traitées par une approche de fouille de texte (bag of words, N-grams) afin de créer une matrice M de n JDOs $\times$ m caractéristiques,
3. Un graphe de proximité (graphe de voisins relatifs, RNG) est calculé à partir de M en utilisant un algorithme parallèle implémenté sur CPU et sur GPU,
4. Le graphe est affiché avec une méthode de dessin de graphe (logiciel Tulip). L'utilisateur peut explorer le graphe d'un point de vue global et peut obtenir grâce aux liens locaux des suggestions.

2.1 Téléchargement et pré-traitement des JDOs

La première étape est réalisée avec un outil de téléchargement de pages web d'un site Open Data (tel que `http://www.data.gouv.fr`), outil que nous avons créé pour nos fins : pour chaque JDO, il extrait les méta-données (titre, mots clés, description) et le contenu du fichier de données. Cette information est re-codé dans un format XML. Notre outil de téléchargement scanne le site Web en entier. Il utilise une implémentation parallèle et une politique respectueuse du serveur (nombre de requêtes limité). Ce téléchargement est incrémental : une fois que la collection complète a été téléchargée et lorsque l'utilisateur demande une mise à jour de cette collection, seuls les nouveaux JDOs ou ceux modifiés sont téléchargés et traités.

2.2 Extraction des caractéristiques

Avant la création des caractéristiques, les méta données et le contenu de chaque JDO sont analysés et convertis en un ensemble de termes ("bag of words") ou de N-grams (méthodes classiques en fouille de texte (Berry, 2004)). Des techniques classiques viennent compléter cette extraction (liste de mots vides, troncation, pondération TFIDF, loi de Zipf). A la fin de cette étape, la matrice M, est créée : chacun des n JDOs est représenté comme un vecteur dans un espace de dimension m.

2.3 Calcul du graphe RNG avec une approche parallèle sur CPU et GPU

Pour construire une carte globale de la collection complète des JDOs ainsi que pour créer des liens locaux entre des JDO similaires, nous avons étudié les graphes de voisinage (Bose et al., 2012). Ces approches peuvent relier les JDOs partageant des similarités importantes. Ces liens locaux peuvent être utilisés pour proposer à un utilisateur en cours d'exploration d'un JDO particulier, d'autres JDOs connexes à explorer. Par ailleurs, le graphe créé peut être visualisé en 2D ou en 3D en utilisant des méthodes de dessin de graphes (Herman et al., 2000). Avec une telle représentation, l'utilisateur peut obtenir une représentation globale de la collection complète des JDOs, ainsi que des détails sur les similarités locales. Par ailleurs, nous avons exclus certaines méthodes comme le MDS (Borg et Groenen, 2005) car le nombre de données à traiter dans nos travaux (voir nos résultats) est au-delà des capacités actuelles du MDS même avec un algorithme optimisé et une implémentation efficace utilisant le GPU (comme dans (Ingram et al., 2009)). Nous avons choisi le graphe des voisins relatif (RNG) (Toussaint, 1980). Sa complexité est élevée ($O(n^3)$), mais il crée un graphe connexe. Ce point est important en termes de visualisation pour positionner en 2D les différents clusters en fonction de leur ressemblance. D'un point de vue local, un graphe connexe garantit qu'un chemin existe entre chaque paire de noeuds, ce qui permet à l'utilisateur d'atteindre n'importe quel JDO lors d'une exploration locale.

Des chercheurs ont tenté de réduire la complexité de cette méthode, comme dans (Hacid et Yoshida, 2007) où une version incrémentale est proposée. Cependant, cette version nécessitait 150 minutes pour traiter $n = 75.000$ données en dimension $m = 50$. Comme nous le verrons dans les résultats, les données que nous traitons sont beaucoup plus volumineuses (environ 64 fois plus grandes : n=300,000 données et m=800 caractéristiques). À notre connaissance, le RNG n'a pas encore été utilisé avec une telle matrice. Ainsi, afin de réduire les temps de traitement, nous avons étudié deux versions parallèles de RNG sur des architectures multi-coeurs (CPU et GPU). Ces approches parallèles construisent les mêmes graphes que l'approche séquentielle. Nous proposons donc dans ce travail la première implémentation de RNG utilisant une architecture multi-coeurs. Nous avons défini deux versions de cette parallélisation : la première version parallèle considère que la matrice de distance M_D entre les n données $d_1, ..., d_n$ peut être entièrement stockée dans la mémoire de l'architecture multi-coeurs. Pour traiter des matrices plus grandes, une deuxième approche a été proposée. Elle utilise une parallélisation plus complexe qui utilise une stratégie parcimonieuse pour calculer les valeurs de distance. Des matrices partielles de distance sont calculées et stockées temporairement.

2.4 Dessin du graphe et exploration interactive

Une fois que le RNG est calculé, il faut construire une visualisation de ce graphe permettant de révéler des clusters ainsi que des relations entre les JDOs. Parmi les méthodes de dessin de graphes, nous avons choisi la représentation noeud-lien, car elle permet de donner une vue d'ensemble du graphe ainsi que des détails locaux, i.e., les liens entre les noeuds. De plus, cette disposition noeud-lien peut être utilisée pour représenter la proximité des noeuds : la longueur (en 2D) des arrêtes affichées $l_{i,j}$ peut dépendre de la distance Euclidienne $Dist(d_i, d_j)$ dans l'espace des caractéristiques. Par ailleurs, la taille importante du graphe que nous voulons traiter (plusieurs centaines de milliers de noeuds et de liens) est un problème nécessitant un algorithme rapide de disposition de graphes (voir une expérience dans (Hachul et Jünger, 2007) où les plus grands graphes testés sont plus petits que ceux que nous traitons dans nos travaux). Par conséquent, nous avons choisi l'algorithme multi-niveaux FM^3 (Hachul et Jünger, 2005).

La représentation visuelle du graphe peut être améliorée en définissant des propriétés pour les noeuds. Par exemple, chaque noeud peut avoir une propriété titre et une propriété URL pour représenter respectivement le titre et l'URL du JDO correspondant. Les titres peuvent être affichés pour révéler à l'utilisateur les thèmes du noeud. L'URL peut être utilisée pour ouvrir le JDO dans un tableur. Nous avons sélectionné un logiciel existant : Tulip 4.1.0 [1]. Cet outil intègre plusieurs méthodes de dessin de graphes (y compris FM^3) et peut créer des visualisations efficaces du graphe. La longueur des arêtes peut être réglée pour représenter le plus fidèlement possible la distance euclidienne entre les noeuds. La couleur des arêtes peut aussi dépendre de cette distance. Tulip offre les interactions standards, y compris la recherche par mots clés dans les propriétés associées aux noeuds. Pour compléter ces interactions, nous avons défini et implémenté un module additionnel dans Tulip permettant de télécharger et d'ouvrir le fichier de données associé à un noeud donné. Ce plug-in utilise la propriété URL associée à chaque noeud et qui, dans notre contexte, représente le lien de téléchargement du fichier de données. Ainsi, lorsque l'utilisateur double-clique sur un noeud, le fichier de données correspondant est ouvert dans un tableur.

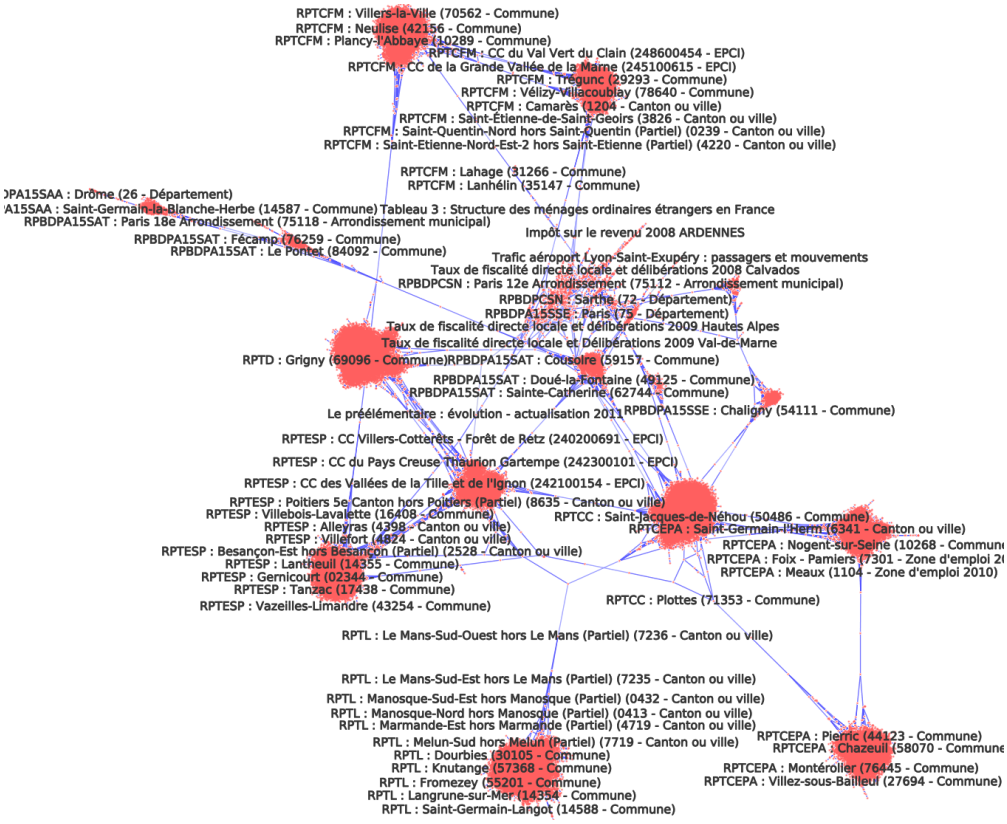
3 Résultats

Nous avons réalisé de nombreux tests dont nous donnons ici un résumé. Les JDOs traités ont été téléchargés à partir `http://www.data.gouv.fr` en Juin 2012. Nous avons sélectionné les JDO dont les fichiers de données étaient au format XLS afin d'être en mesure de traiter leurs contenus. Les méta-données de chaque JDO ainsi que le contenu textuel du fichier de données ont été stockés au format XML. Le nombre de JDOs s'élevait à $n = 293769$.

Pour traiter la collection complète et calculer à partir de celle-ci le graphe RNG, seule l'extraction de 2-grams permet d'obtenir un nombre de descripteurs suffisamment réduit et compatible avec le calcul de RNG. Lors de la phase d'extraction des caractéristiques, nous avons obtenu $m = 793$ descripteurs pour la collection complète et en exploitant toutes les sources de données (méta-données et contenu). Nous avons construit RNG sur cette matrice de données, et nous avons obtenu un graphe ayant 715 709 arêtes en 19H de calcul environ. Cependant, lors de la visualisation de ce graphe (293 769 noeuds, 715 709 arêtes), les performances de Tulip sur notre matériel n'ont pas permis d'obtenir un résultat satisfaisant. En

1. `http://tulip.labri.fr`

FIG. 1 – *Le graphe construit avec $n = 146928$ JDOs avec l'approche "2-Grams" pour l'extraction des caractéristiques. Ce graphe a 342559 arêtes. Chaque noeud est un JDO.*



conséquence, pour obtenir malgré cela une représentation visuelle du graphe, nous avons pris un échantillon aléatoire composé de la moitié de la collection totale des JDOs. Avec deux fois moins de noeuds (146 928), nous avons construit le graphe RNG (342 559 arêtes et moins de 3h de temps de calcul) et nous avons pu le visualiser (voir figure 1).

A partir de ce premier résultat, nous avons effectué les expériences suivantes (que nous présenterons lors de l'atelier) :

- nous avons analysé les clusters découverts par notre approche, à la fois d'un point de vue global (relation entre les clusters) et local (sous-clusters). Leur contenu s'est révélé pertinent et représentatif,
- nous avons analysé une sous-partie des JDOs, ceux concernant les données qui ne sont pas de recensement. En effet, notre graphe a montré visuellement que la collection globale est dominée par les données de l'INSEE, alors qu'une partie beaucoup plus réduite traite de sujets variés et très intéressants (loisirs, sécurité, ...). A nouveau, pour cet ensemble plus réduit, le positionnement 2D des noeuds a été jugé comme pertinent par

rapport au contenu des JDO qu'ils représentent,
– nous avons étudié l'impact, en termes de perte d'information, de l'utilisation des 2-grams. Pour cela nous avons construit différents graphes avec les 2-grams et ce jusqu'aux 6-grams et aussi avec l'approche bag of word. Nous avons montré avec des indices de similarité que les graphes construits sont très similaires et que la perte d'information est limitée,
– nous avons montré comment le graphe et ses liens locaux peuvent être utilisés pour suggérer des JDOs à explorer. A partir d'une requête classique (présence de mots clés dans les titres des JDO) retournant un premier ensemble de noeuds, le résultat de la requête peut être étendus aux sous-graphes portés par ces noeuds initiaux contenant les mots clés. Nous avons alors montré, sur quelques exemples, que les liens établis sont pertinents (des JDO effectivement similaires et intéressants sont suggérés à l'utilisateur). Nous avons également montré qu'en terme de précision, cette approche utilisant les liens locaux est meilleure, pour des requêtes spécifiques, que le moteur de recherche présent sur le site Open Data français.

4 Conclusions et perspectives

Nous avons présenté nos travaux sur notre outil permettant l'exploration d'une grande collection de JDOs. Les principales caractéristiques de notre approche sont les suivantes : des JDOs sont téléchargés à partir d'un site Web d'Open Data et en sont extraits les méta-données et le contenu textuel. Des techniques de fouille de texte sont utilisées pour définir les caractéristiques représentant les JDOs dans un espace vectoriel. Un graphe RNG est construit en créant des liens entre les JDOs en fonction de leurs similarités. Une implémentation GPU de l'approche RNG a été utilisée pour traiter une grande matrice de données. Le logiciel Tulip a servi d'interface utilisateur pour visualiser le graphe et offrir aux utilisateurs des interactions leur permettant d'explorer la collection.

Plusieurs expériences ont été menées avec les données de l'Open Data français. Compte tenu de la grande taille de la collection, seule l'approche "2-Grams" a pu être adaptée pour construire un graphe à partir de la collection complète de JDOs. Nous avons montré que la perte d'informations avec "2-Grams" reste faible et n'a pas d'impact important sur les graphes créés. Nous avons montré l'efficacité de l'approche GPU. En raison de la limitation de la visualisation des graphes dans Tulip, nous n'avons pu visualiser que la moitié de cette collection (mais cette visualisation peut sans doute être améliorée). Notre visualisation a confirmé que le graphe RNG de Toussaint peut révéler des clusters de JDOs. Avec un graphe réduit aux données hors-recensement, nous avons utilisé l'approche "bag of words" pour l'extraction des caractéristiques. Nous avons construit le RNG correspondant. La visualisation de ce graphe a révélé des clusters intéressants. En ce qui concerne l'évaluation des liens créés, nous avons réalisé une expérience simple qui a mis en évidence la qualité de ces liens.

Nous pouvons donner plusieurs perspectives au travail présenté dans ce chapitre. Nous aimerions utiliser un algorithme parallèle sur GPU pour le dessin du graphe afin de gérer la collection complète de JDOs. Cela représenterait une quantité importante de travail car il faudrait complètement redéfinir l'interface utilisateur. Une seconde perspective consiste à fournir plus d'aide aux utilisateurs. Notre outil est capable de représenter visuellement les résultats et les suggestions sous forme d'un graphe interactif. Pour fournir davantage d'aide aux utilisateurs,

nous pensons qu'avec un assistant de l'utilisateur tel que (Healey et al., 2008), l'utilisateur peut obtenir automatiquement une visualisation du jeu de données sélectionné. Avec un tel assistant, notre outil serait totalement interactif et visuel, de la représentation globale de la collection à l'exploration détaillée du contenu d'un jeu de données. Enfin, nous aimerions publier les résultats du graphe dans un site Web. Avec un navigateur Web, visualiser la collection complète est impossible car le client n'aurait pas de CPU ou de GPU assez puissant. Cependant, les sous-graphes correspondant à une requête particulière peuvent être visualisés sur un site web. L'assistant utilisateur pourrait également être implémenté dans ce site web.

Références

Berry, M. W. (2004). *Survey of Text Mining I : Clustering, Classification, and Retrieval*, Volume 1. Springer.

Borg, I. et P. Groenen (2005). *Modern multidimensional scaling : theory and applications, Second edition, Series in Statistics*. Springer.

Bose, P., V. Dujmovic, F. Hurtado, J. Iacono, S. Langerman, H. Meijer, V. S. Adinolfi, M. Saumell, et D. R. Wood (2012). Proximity graphs : E, δ, Δ, χ and ω. *Int. J. Comput. Geometry Appl. 22*(5), 439–470.

Hachul, S. et M. Jünger (2005). Large-graph layout with the fast multipole multilevel method. Technical report.

Hachul, S. et M. Jünger (2007). Large-graph layout algorithms at work : An experimental study. *Journal of Graph Algorithms and Applications 11*(2), 345–369.

Hacid, H. et T. Yoshida (2007). Incremental Neighborhood Graphs Construction for Multidimensional Databases Indexing. In *the Canadian Artificial Intelligence Conference, CanAI 2007, LNAI 4509*, Montreal, Quebec, Canada, pp. 405–416.

Healey, C., S. Kocherlakota, V. Rao, R. Mehta, et R. St Amant (2008). Visual perception and mixed-initiative interaction for assisted visualization design. *IEEE Transactions on Visualization and Computer Graphics 14*(2), 396–411.

Herman, I., G. Melancon, et M. S. Marshall (2000). Graph visualization and navigation in information visualization : A survey. *IEEE Transactions on Visualization and Computer Graphics 6(1)*, 24–43.

Ingram, S., T. Munzner, et M. Olano (2009). Glimmer : Multilevel MDS on the GPU. *IEEE Transactions on Visualization and Computer Graphics 15*, 249–261.

Shneiderman, B. (1996). The eyes have it : A task by data type taxonomy for information visualizations. In *Proceedings of the 1996 IEEE Symposium on Visual Languages*, VL '96, Washington, DC, USA, pp. 336–343. IEEE Computer Society.

Toussaint, G. T. (1980). The relative neighborhood graphs in a finite planar set. In *Pattern recognition*, Chapter 12, pp. 261–268.

Construction et exploration d'un graphe de jeux de données publiques

Summary

We summarize in this paper our work on the building of a visual and interactive map of a large collection of Open datasets (a paper will soon be published in an international journal). Our approach uses 1) text mining techniques to define a representation space for such data, 2) the building of a proximity graph with a CPU and GPU based parallel approach, 3) the interactive visualization of this graph with the Tulip Software. We performed several tests to highlight the limitations and the properties of our approach.

Démonstration d'une visualisation radiale pour l'exploration de grands volumes de données

Tianyang Liu*, Fatma Bouali**,*, Gilles Venturini*

* Université François-Rabelais de Tours, Laboratoire d'Informatique
64 avenue Jean Portalis, 37200 Tours, France
tianyang.liu@etu.univ-tours.fr, venturini@univ-tours.fr
** Université de Lille2, IUT, Dpt STID
25-27 Rue du Maréchal Foch, 59100 Roubaix, France, fatma.bouali@univ-lille2.fr

Résumé. Dans cet article, nous présentons une démonstration d'une méthode permettant de visualiser de grands volumes de données multidimensionnelles. Nous rappelons brièvement les principes de notre méthode : des points d'intérêt sont choisis parmi les données et disposés sur un cercle (typiquement de 5 à 10 points). Les données restantes sont affichées en fonction de leur ressemblance à ces points d'intérêt. Une parallélisation sur CPU et GPU permet de visualiser avec des temps très courts des millions de données sur des dizaines de dimensions et permet d'avoir des interactions très rapides. Nous comparons notre méthode avec les visualisations obtenues avec d'autres approches, lorsque le volume de données le permet.

1 Problématique abordée

Nous considérons un jeu de n données multidimensionnelles $d_1, ..., d_n$ où chaque donnée est décrite par m attributs $A_1, ..., A_m$. A titre indicatif, nous considérons dans cette démonstration des données telles que $n * m$ soit de l'ordre de 130 10^6. On peut distinguer au moins quatre manières d'étendre les méthodes visuelles et interactives pour le traitement de masses de données multidimensionnelles : la visualisation, les interactions, la réduction des temps de calcul via une amélioration de la complexité algorithmique et/ou via la parallélisation. En ce qui concerne la visualisation, on peut dire que les approches pouvant afficher le plus de données possible, relèvent soit des approches orientées pixels et assimilées (1 pixel étant la plus petite unité visualisable), soit des approches utilisant des densités (la transparence donne une information sur la masse de données présentes). Dans les deux cas, l'affichage d'un grand ensemble de données peut aussi impliquer du matériel spécifique (mur d'écrans par exemple). Parmi les approches orientées pixels et assimilées, on peut citer (Keim et al., 1995), où 530.000 valeurs sont visualisées, ou encore les TreeMaps qui ont été étendus pour représenter jusqu'à 1 million de noeuds (Fekete et Plaisant, 2002). Parmi les approches utilisant des densités, on peut citer (Fua et al., 1999) où les coordonnées parallèles peuvent afficher jusqu'à 200.000 données. En ce qui concerne les interactions, on peut constater que c'est le point d'achoppement des méthodes visuelles pour le traitement de grands volumes de données (voir discussion dans

(Florek, 2006)). Lorsque l'utilisateur formule une requête graphique, le système doit bien souvent faire des calculs supplémentaires et réafficher les données, partiellement ou entièrement. Les méthodes énumérées précédemment gèrent mal les interactions pour les grands volumes de données. Elles peuvent pratiquement être considérées comme statiques puisque les temps d'affichage sont souvent longs : le système dispose d'autant de temps que nécessaire pour calculer la visualisation avant de la montrer à l'utilisateur, mais une redéfinition interactive de la visualisation au cours d'une session utilisateur impose une contrainte de temps encore plus forte que pour la définition d'une visualisation "fixe".

En ce qui concerne la complexité algorithmique et la parallélisation des méthodes visuelles, les approches concernant le Multi-Dimensional scaling (MDS) sont représentatives des travaux effectués. Pour le MDS (voir détails dans (Ingram et al., 2009)), un premier effort dans la communauté porte sur la diminution de la complexité algorithmique, mais cette complexité est encore importante si l'on veut traiter des millions de données (au delà du linéaire). Un deuxième effort concerne ainsi la parallélisation, et notamment sur GPU, comme dans (Ingram et al., 2009), mais il s'agit d'une approche peu interactive (sans redéfinition de l'espace de visualisation). Le même principe de parallélisation a été appliqué aux coordonnées parallèles dans (Florek, 2006) avec cette fois comme objectif d'obtenir des temps d'affichage permettant les interactions. Le facteur d'accélération entre l'implémentation sur CPU (non parallélisée) et l'implémentation sur GPU va jusqu'à 60 (150.000 données en dimension 4 sont affichées en 0.3s sur GPU au lieu de 20s sur CPU). Enfin, un autre type d'approche parallèle sur GPU concerne les cartes de Kohonen qui donnent en sortie un résultat visuel (Zhongwen et al., 2005). Cependant, ces approches ne visualisent pas directement les données mais plutôt des clusters.

2 Description de la méthode et démonstration

Pour ces raisons, nous nous sommes tournés vers une méthode qui soit de complexité faible et qui puisse se paralléliser sur CPU/GPU. Notre but était de répondre du mieux possible aux trois critères (visualisation, interactions, complexité) et de permettre à l'utilisateur de réaliser des tâches de découvertes de clusters (détection de groupes, d'outliers, et d'autres informations topologiques, découverte interactive du contenu des clusters, etc). L'approche choisie fait partie des méthodes radiales (voir survol dans (Draper et al., 2009)) avec comme exemple représentatif la méthode RadViz (Hoffman et al., 1999). Ces approches radiales considèrent que des points d'intérêt (POIs, appelés aussi "ancres dimensionnelles"), sont disposés sur un cercle par exemple, et que les données viennent se positionner à l'intérieur du cercle en fonction de leur ressemblance avec les POIs (voir figure 1). Ainsi, les POIs peuvent représenter des cas particuliers importants parmi les données, ou des hypothèses à tester ou encore des concepts. Chaque choix ou configuration des POIs vient donner une disposition particulière des données. L'utilisateur dispose de plusieurs interactions afin de modifier cette disposition et faire apparaître de nouvelles informations. Nous ne détaillerons pas plus les propriétés de ces approches en matière de visualisation et de fouille de données (voir détails par exemple dans (Daniels et al., 2012)).

Notre approche peut être résumée ainsi. Notons $POI_1, ..., POI_k$ les k points d'intérêt choisis. Une session utilisateur se déroule selon l'algorithme suivant découpé en 5 étapes :

E1 Lecture des données $d_1, ..., d_n$

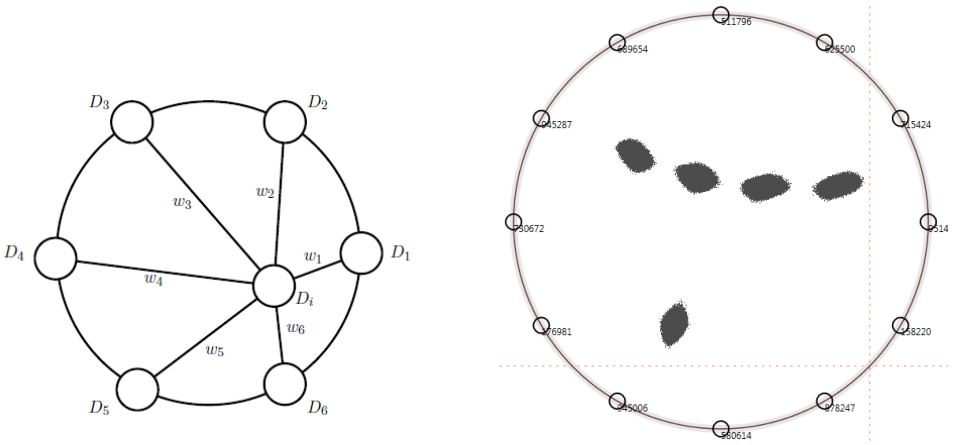


FIG. 1 – *Principe de la méthode d'affichage radial à base de points d'intérêt (à gauche) : chaque donnée est positionnée en fonction de sa similarité avec les points d'intérêt qui sont disposés autour du cercle. A droite nous donnons un exemple d'affichage d'une base articifielle (1 million de données et 10 attributs) dont l'obtention (calcul et affichage) prend 123ms.*

E2 Définition initiale des points d'intérêt POI_1 à POI_k en utilisant un algorithme heuristique

E3 Calcul des similarités entre les données et les POIs (matrice $n * k$) et des coordonnées d'affichage des données

E4 Affichage des données

E5 Interaction avec l'utilisateur et traitement de la requête graphique demandée (avec un retour en E3 pour un changement de POIs, ou un retour en E4 pour un zoom ou une sélection).

Les différentes étapes de cet algorithme ont été parallélisées (notamment l'etape E3 sur qui a lieu sur GPU, les autres étant parallélisées sur le CPU). Egalement, nous avons inclus de nombreuses interactions : l'utilisateur peut zoomer, ajouter/enlever des POIs, transformer une donnée en POI (et réciproquement), sélectionner des données, les étiqueter ou encore en obtenir une description conceptuelle. En présence de masse de données, ces opérations de visualisation et d'interaction deviennent complexes et doivent être traitées de manière rapide afin de ne pas ralentir ou décourager l'utilisateur dans son processus d'exploration. Notre outil peut traiter toutes ces interactions en moins d'une seconde pour des données telles que $n*m \leq 130\ 10^6$ (avec un processeur i5 et un GPU Nvidia QuadroFX6000).

Nous effectuerons une démonstration de notre méthode sur un portable équipé d'un GPU NVidia. Pour utiliserons les différents jeux de données présents sur le site de l'atelier. Nous montrerons, pour ces mêmes jeux de données et si leur taille le permet, les résultats que l'on peut obtenir avec d'autres approches classiques (projections linéaires et non linéaires) afin de bien mettre en lumière les avantages et inconvénients de notre méthode.

Références

Daniels, K., G. Grinstein, A. Russell, et M. Glidden (2012). Properties of normalized radial visualizations. *Information Visualization 11*(4), 273–300.

Draper, G. M., Y. Livnat, et R. F. Riesenfeld (2009). A survey of radial methods for information visualization. *IEEE Trans. Vis. Comput. Graph. 15*(5), 759–776.

Fekete, J. et C. Plaisant (2002). Interactive Information Visualization of a Million Items Proceedings of IEEE Symposium on Information Visualization.

Florek, M. (2006). Using modern hardware for interactive information visualization of large data. *Master's thesis, Comenius University, Bratislava*.

Fua, Y.-H., M. O. Ward, et E. A. Rundensteiner (1999). Hierarchical parallel coordinates for exploration of large datasets. In *Proceedings of the conference on Visualization '99 : celebrating ten years*, VIS '99, pp. 43–50. IEEE Computer Society Press.

Hoffman, P., G. G. Grinstein, et D. Pinkney (1999). Dimensional anchors : A graphic primitive for multidimensional multivariate information visualizations. In *Workshop on New Paradigms in Information Visualization and Manipulation'99*, pp. 9–16.

Ingram, S., T. Munzner, et M. Olano (2009). Glimmer : Multilevel MDS on the GPU. *IEEE Transactions on Visualization and Computer Graphics 15*, 249–261.

Keim, D. A., M. Ankerst, et H. P. Kriegel (1995). Recursive Pattern : A Technique for Visualizing Very Large Amounts of Data. *Proceedings of the 6th conference on Visualization'95*, 279–286.

Zhongwen, L., L. Hongzhi, Y. Zhengping, et W. Xincai (2005). Self-Organizing Maps computing on Graphic Process Unit. In *13th European Symposium on Artificial Neural Networks (ESANN)*, pp. 557–562. d-side.

Summary

In this paper, we present a demonstration of a method that can visualize large amounts of multidimensional data. We briefly describe the main principles of our approach: several points of interest are selected among the data and are laid out on a circle (from 5 to 10 points). The remaining data are displayed at a location that depends on the similarities between the data and the points of interests. A CPU/GPU based parallel approach is used to visualize in a few seconds millions of data with tens of dimensions and to support quick interactions even for large datasets. We compare our approach to other existing ones when the volume of data allows us to do so.

Traitement Visuel et Interactif dans le Logiciel Cadral

Philippe Pinheiro, Yoann Didry, Olivier Parisot Thomas Tamisier

Centre de Recherche Public - Gabriel Lippmann
41, rue du Brill, L-4422 Belvaux, Luxembourg
tamisier@lippmann.lu

Résumé. Mise au point grâce à de nombreux partenariats de recherche et développement, la plate-forme logicielle Cadral répond aux besoins opérationnels des organisations par l'intégration de deux modules fortement complémentaires : d'une part, un système d'aide à la décision conçu pour le travail collaboratif, et d'autre part, un module d'extraction de connaissance et de traitement de données de type "Visual Analytics". Cet article décrit les fonctionnalités principales de Cadral en traitement visuel de données, et sert de support à une démonstration de Cadral sur les données multidimensionnelles mises à disposition lors de l'Atelier EGC-VIF 2014.

1 Introduction

Cadral offre tout d'abord une infrastructure générale de résolution automatique de problèmes, couplée à des fonctionnalités d'interaction et de modélisation de l'expertise sur un domaine. Conçu prioritairement pour des applications professionnelles dans le contexte des organisations, Cadral entend faciliter au maximum le travail de l'utilisateur, en particulier pour la modélisation de l'expertise métier. Aussi a-t-il été complété par un module d'extraction de connaissance permettant la construction semi-automatique d'un modèle décisionnel à partir d'un historique de données documentant des processus métiers. Ce module, à son tour, est à l'origine d'un ensemble de fonctionnalités permettant le traitement visuel des données avant ou pendant leur exploitation dans une architecture de raisonnement automatique.

Les travaux aspects les plus significatifs du module de traitement visuel sont les suivants.

1. L'analyse de données assistée par la visualisation, permettant par exemple l'affinage de résultats ou le guidage d'utilisateurs en fonction de leurs préférences. Cadral permet de visualiser un même jeu de données selon différents critères : résultats de simulation, voisinage des données, règles extraites…

2. Un ensemble de vues permettant de suivre interactivement des traitements classiques de nettoyage, correction, sélection, clustering... sur diverses structures de

représentation des données : arbres de décisions, projections planaires de données multidimensionnelles, séries temporelles, heatmap permettant le tri et la classification, 'table lens'…

3. Une algorithme original de classification : DTA (DecisionTree Aware Clustering). Les méthodes disponibles de classification utilisent des critères essentiellement intrinsèques aux données (leur voisinage). La particularité de DTA et de classifier les données en vue d'optimiser leur représentation. Exemples d'application : (i) visualisation d'un grand jeu de données en parties compactes, (ii) en raisonnement automatique, décomposition d'un modèle décisionnel complexe en modules plus simple à appréhender.

2 Scénario de démonstration

Nous donnons ci-dessous les grandes étapes d'une démonstration typique, telle que nous proposons de faire sur les données multidimensionnelles mises à disposition pour l'atelier VIF-EGC 2014.

La modélisation

Permet la création et l'édition d'arbres de décision, ainsi que leur test, leur simulation, et la réorganisation automatique des arbres.

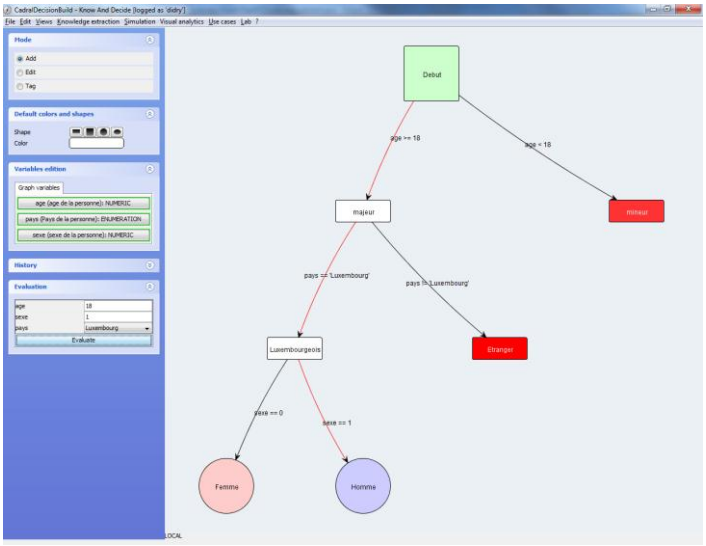


FIG. 1 – *Interface de modélisation*

Extraction depuis une table de données

Permet d'extraire la connaissance comprise dans une table de données en créant un arbre de décision par application de l'algorithme C4.5 [3], les états finaux étant les valeurs d'un attribut choisi comme classe de décision. Il est possible d'obtenir un compromis entre exactitude du modèle et simplicité de l'arbre en jouant avec le *Confidence Factor* (CF faible : arbre plus simple et moins exact).

FIG. 2 – *Paramétrage du Confidence Factor*

Les deux illustrations suivantes montrent le même arbre de décision avec des CF différents.

FIG. 3 – *Arbre le plus exact possible*

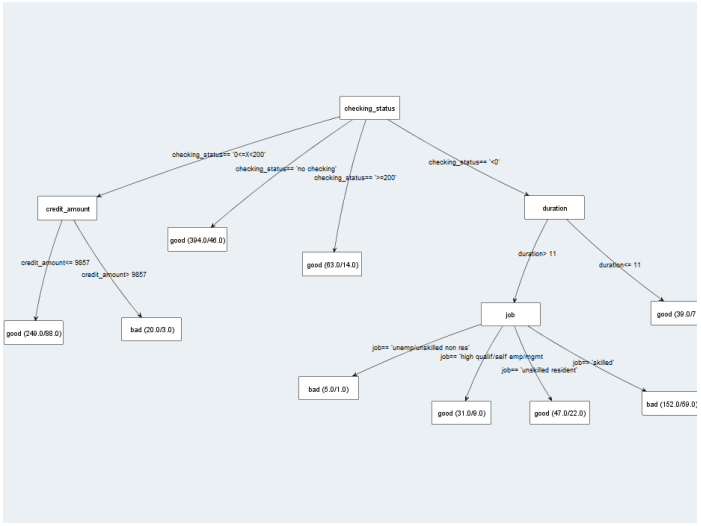


FIG. 4 – *Arbre simplifié*

Use-cases

Cadral permet de préparer des cas pratiques d'utilisation (*use-cases*) des arbres de décision comme moteur de classification ou de prédiction, ainsi que pour trouver des corrélations entre les données ou leurs attributs. Les use-cases sont particulièrement intéressants en vue de la traçabilité des résultats et de l'édition facilitée de rapports. Les résultats des arbres de décision sont complétés par d'autres techniques (notamment les plus proches voisins) afin de nuancer les résultats. Les illustrations suivantes montrent deux exemples d'use-cases.

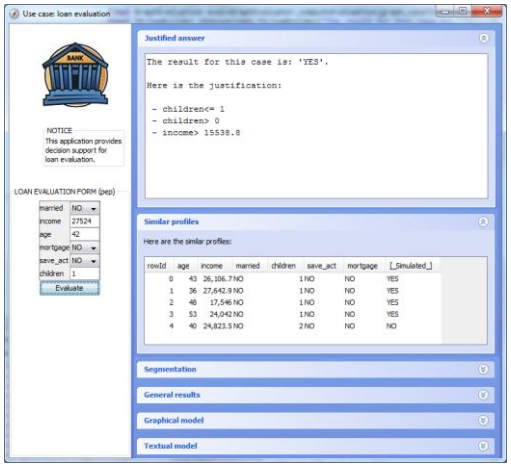


FIG. 5 – *Acceptation de prêt bancaire*

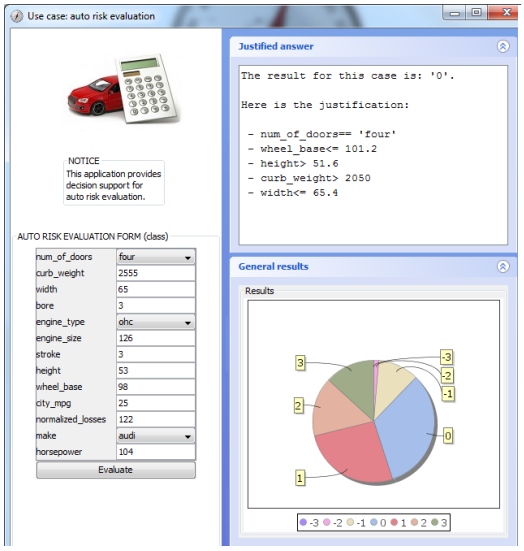


FIG. 6 – *Calcul du risque pour une assurance auto*

Visualisation interactive et exploration

Cadral offre différentes vues interactive, se mettant mutuellement à jour instantanément pour guider l'exploration de données, en particulier celles montrées sur la figure suivante (tableaux, arbres de décision, vues statistiques : vues textuelles + sommaire par attribut).

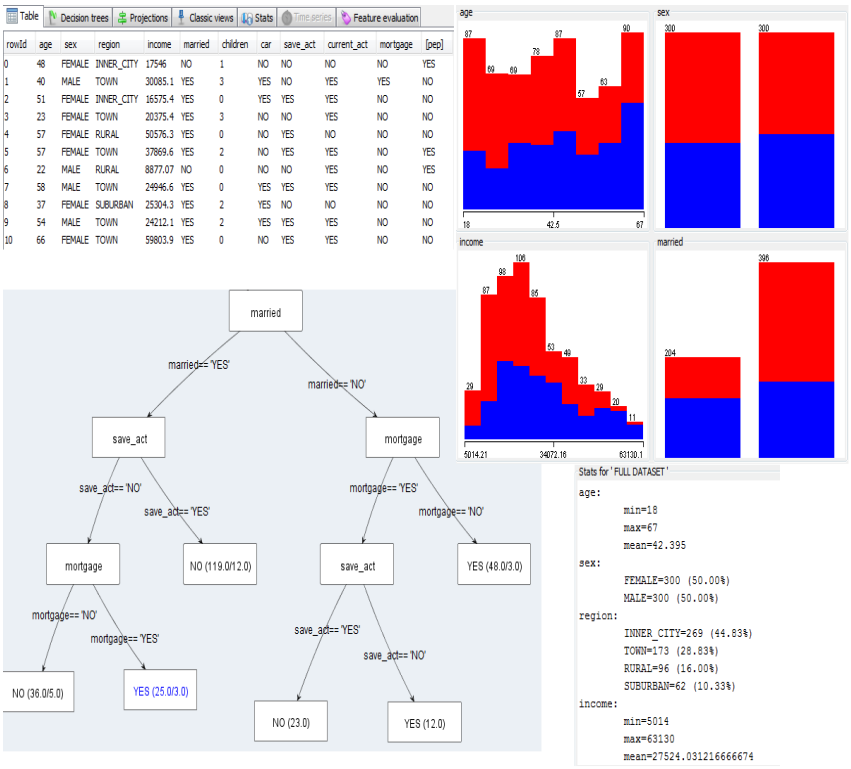


FIG. 7 – *Exploration de jeux de données*

Projections

Cadral permet l'utilisation de projections permettant de réduire la dimensionnalité des données pour leur représentation, comme illustré par les deux figures suivantes. La projection en composante principales (PCA) met en évidence les liens entre les attributs, et sert à compresser un ensemble de dimension quelconque en choisissant les 2 meilleurs axes qui expliquent la variance. La projection *Multi-Dimensional Scaling* (MDS) est une réduction planaire préservant la distance entre les éléments : deux éléments qui proches dans Rn seront proches dans R^2.

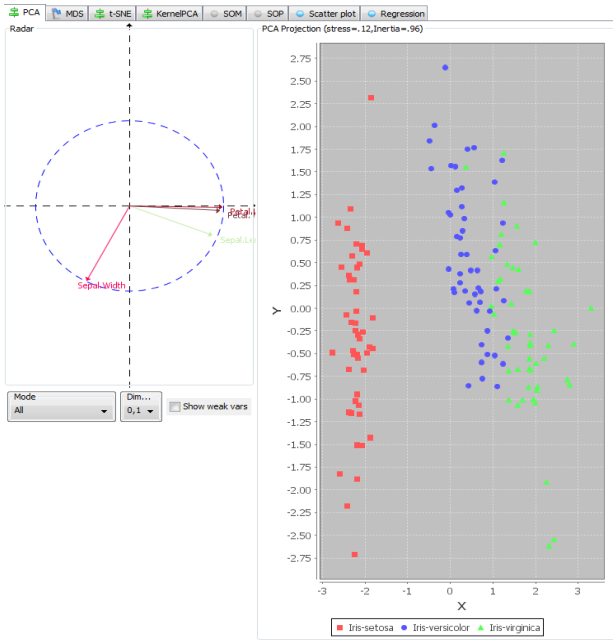


FIG. 8 – *Projection PCA*

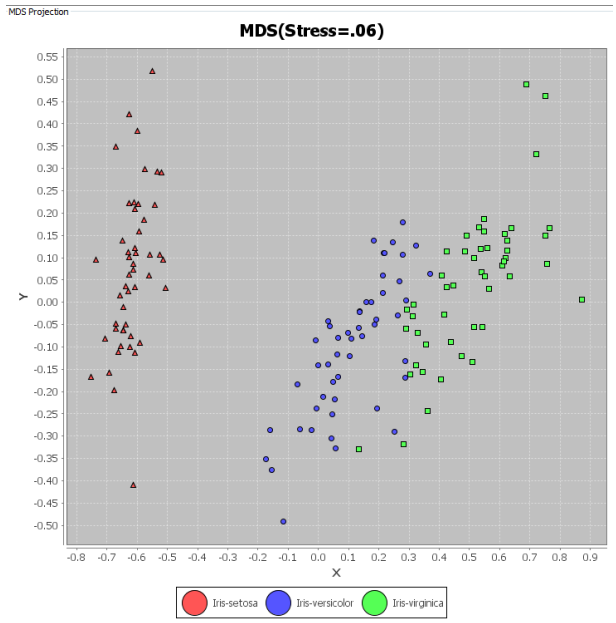


FIG. 9 – *Projection MDS*

Vues originales

Des vues originales permettent de mettre en évidence les corrélations entre des attributs de toute nature. Au préalable, les attributs symboliques sont numérisés, puis l'ensemble des attributs sont normalisés et associés à un code de couleur. Les illustrations suivantes montrent deux vues originales. Dans la *heat-map*, chaque ligne est représentée par un ensemble de pixels, et l'intensité de la couleur dépend de la valeur de l'attribut. Dans la *table lens*, chaque attribut d'une donnée est représenté par un trait de taille proportionnelle à sa valeur.

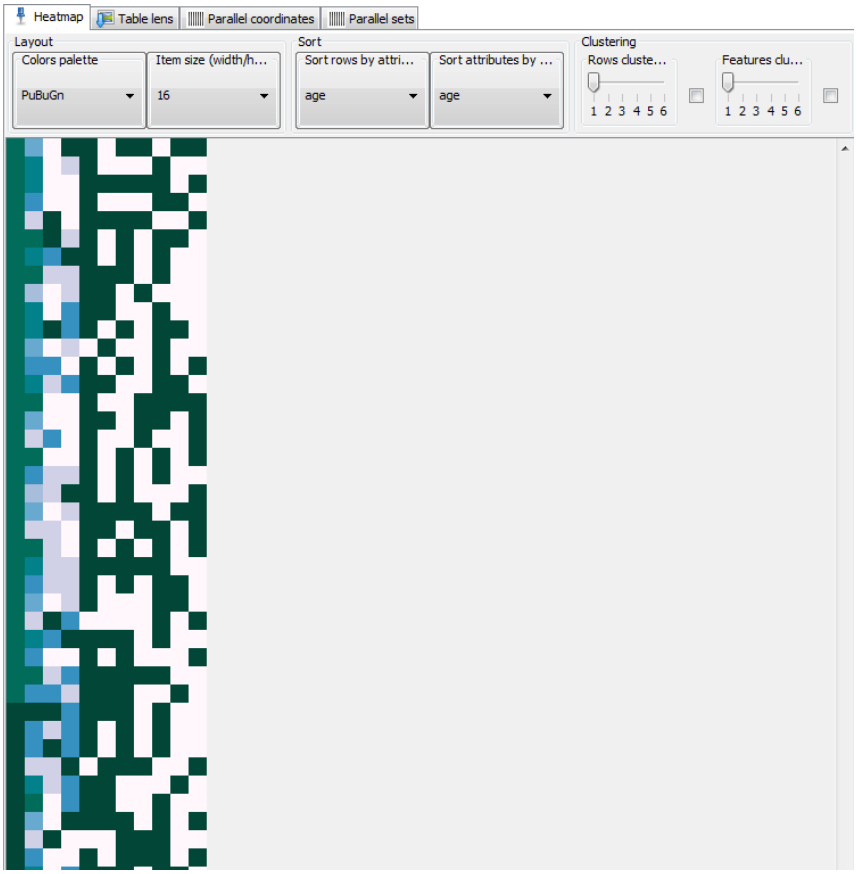


FIG. 10 – *Heatmap*

Traitement visuel interactif dans le logiciel Cadral

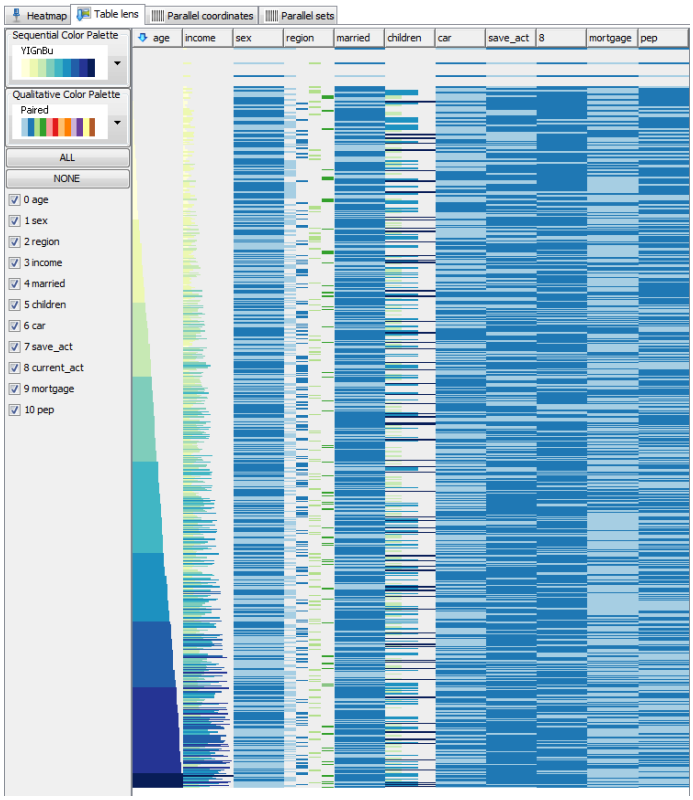


FIG. 11 – *Table lens*

Filtrage interactif des données

Divers filtres sont disponibles pour affiner et explorer interactivement des ensemble de données. Parmi ceux-ci, la hiérarchisation réordonne les colonnes par importance des attributs dans l'arbre de décision, la simplification supprime les attributs ayant peu d'importance), la discrétisation convertit les valeurs numériques en intervalles.

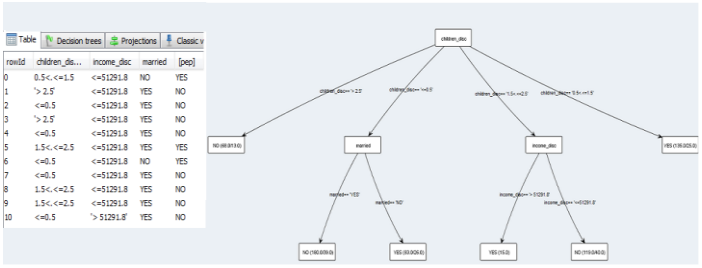


FIG. 12 – *Arbre simplifié interactivement*

Traitement de données interactif

Différents types d'opérations sont possibles comme la réduction du nombre de dimensions, la réduction du nombre d'éléments, la transformation des données (discrétisation, numérisation, transposition…), le nettoyage des données (valeurs extrêmes, bruit).

Le traitement exploratoire permet de revenir aux données d'origine ou intermédiaires et ainsi de jouer avec les données par essais et erreurs, et un travail récent a été proposé pour automatiser ce processus [2]. Il est également possible d'exporter le jeu de données 'transformé' dans divers formats.

Un indicateur visuel mesure l'altération du jeu de données.

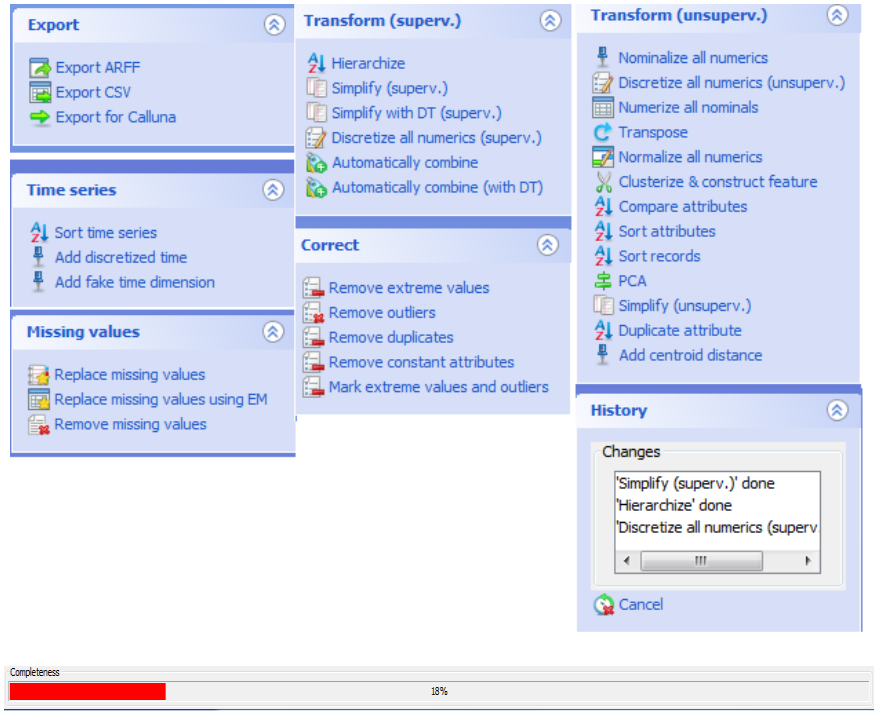


FIG. 13 – *Interface de traitement avec indicateur d'altération*

Clustering

Les jeux de données sont souvent difficiles à étudier en raison de leur taille. Le *clustering* a ainsi pour but de les découper de manière pertinente. Classiquement, le

clustering considère une distance liée aux données elles-mêmes pour regrouper les données proches dans un même cluster. Nous proposons un clustering original, DTA (pour *Decision Tree Aware Clustering*), qui a pour but de simplifier la représentation des données dans une structure au choix [1], afin par exemple de découper un modèle décisionnel en modules élémentaires, ou de faciliter la visualisation d'un grand jeu de données. Les deux illustrations suivantes montrent chacune le résultat d'un clustering créant 2 clusters sur le même jeu de données. Les clusters sont représentés par leurs arbres de décision. Les arbres de la première figure sont obtenus par un clustering basé sur la distance Kmeans. Ceux de la seconde figure, plus simples, sont obtenus par l'algorithme DTA.

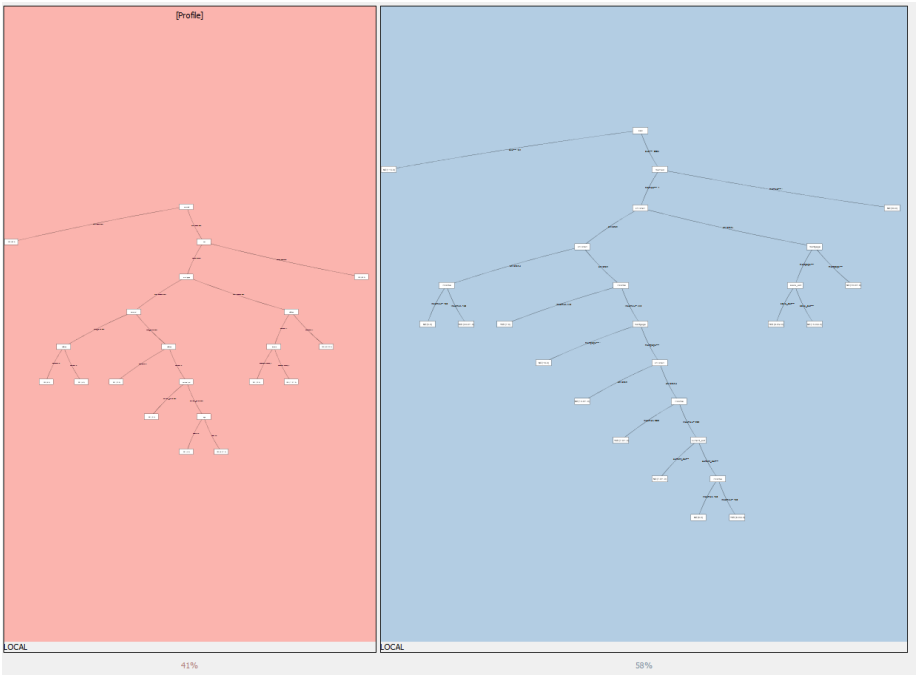


FIG. 14 – *Clustering employant Kmeans*

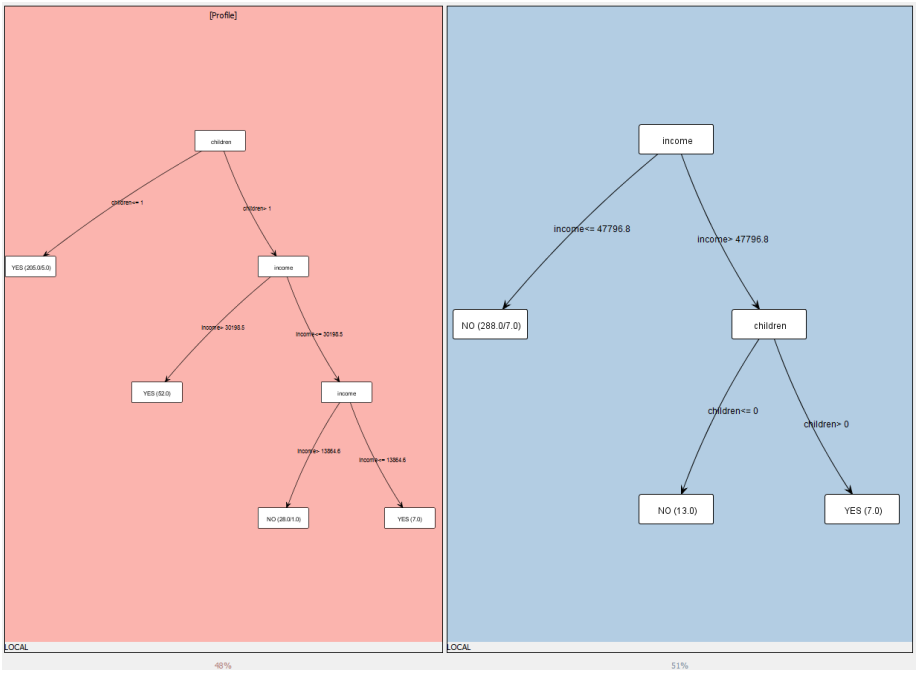


FIG. 15 – *DTA Clustering*

3 Conclusion

Les fonctionnalités passées en revue ci-dessus forment la base des démonstrations classiques du logiciel Cadral. Nous proposons de réaliser une telle démonstration sur les données multidimensionnelles mises à disposition pour l'atelier VIF-EGC2014. Nous souhaitons également à l'occasion de cet atelier discuter des nouvelles perspectives de Cadral concernant les problématiques relatives aux données à composantes temporelles, que nous abordons notamment dans un projet d'analyse de données de capteurs environnementaux.

Références

[1] O. Parisot, Y. Didry, T. Tamisier, B. Otjacques, «Using clustering to improve decision trees visualization», Information Visualisation (IV), 2013 17th International Conference on, 15-18 July 2013, London, UK, pp 186-191
[2] O. Parisot, P. Bruneau, Y. Didry, T. Tamisier: « User-driven data preprocessing for decision support », 10th International Conference on Cooperative Design, Visualization and Engineering, September (CDVE 2013), 22-25, September, 2013, Mallorca, Spain », paru dans « Lecture Notes in Computer Science 8091 », pp. 81-84
[3] J. Ross Quinlan. C4.5: Programs for Machine Learning. Morgan Kaufman, 1993

Traitement visuel interactif dans le logiciel Cadral

Summary

Developed through many industrial and research partnerships, the software platform Cadral answers operational needs of organizations by integrating two complementary modules: on the one hand, a collaborative decision support framework, and on the other hand, a visual analytics tool suite for knowledge extraction and data processing. In this article we introduce the main functionalities about visual data mining, which will be demonstrated on data available during the EGC-VIF 2014 Workshop.

Un système de clustering visuel et interactif

Pierrick Bruneau, Philippe Pinheiro, Bertjan Broeksema et Benoît Otjacques

Centre de Recherche Public - Gabriel Lippmann, 41, rue du Brill, L-4422 Belvaux
(bruneau | pinheiro | broeksem | otjacque)@lippmann.lu

Résumé. Cet article décrit brièvement un système de clustering visuel interactif en cours de développement. À partir d'une projection 2D et d'un clustering calculés initialement, notre contribution consiste à donner les moyens à un utilisateur de modifier interactivement et itérativement cette représentation initiale. L'utilisateur peut ainsi injecter sa connaissance directement dans le clustering, voire corriger visuellement d'éventuelles erreurs. Il peut aussi utiliser la technique pour structurer sa carte mentale des données traitées.

Ce travail en cours émerge de deux besoins complémentaires relatifs au clustering. Tout d'abord, nous avons identifié celui de visualiser efficacement les résultats d'un algorithme de clustering. Turkay et al. (2011) ont notamment proposé une méthode d'évaluation visuelle d'ensembles de clusterings. Ce type de méthode permet notamment de souligner les structures stables dans de tels ensembles. Toutefois, cette approche se détache sciemment des données sous-jacentes, en se concentrant sur les modalités de clusters. Dans la même veine, Seo et Shneiderman (2002) ont proposé une méthode d'inspection et de comparaison visuelle de structures de clusters hiérarchiques.

Ces méthodes sont efficaces pour inspecter et évaluer des collections de clusterings statiques, mais ne permettent pas de changer ou d'adapter des structures de clusters existantes. Or il nous semble qu'une approche de clustering semi-automatique, supportée par des moyens visuels et interactifs, peut être utile à un utilisateur, par exemple pour injecter une connaissance experte dans un clustering a priori, ou faciliter la formation d'une carte mentale d'un jeu de données et du clustering associé.

Des contributions ont été effectuées dans ce domaine, par exemple Rinzivillo et al. (2008), Andrienko et al. (2009), ou encore Lee et al. (2012). Ces dernières implémentent en quelque sorte un paradigme de *data mining visuel*, où l'inspection des résultats de clustering guide le paramétrage de l'algorithme sous-jacent.

Plus en ligne avec l'approche de Philippeau et al. (2008), nous proposons un système permettant de modifier, d'adapter, un résultat de clustering de manière interactive, sans référence explicite à un algorithme de clustering particulier. Notre approche est résumée par la figure 1.

En tant qu'initialisation, nous utilisons t-SNE (van der Maaten et Hinton (2008)), pour réduire la dimensionalité d'un jeu de données arbitraire, et permettre sa représentation en 2D. Par ailleurs, cette méthode de projection souligne les structures de similarité locales entre éléments, et est, à ce titre, souvent utilisée en tant que pré-processing pour réaliser le clustering de jeux de données à grande dimensionalité (voir par exemple Gisbrecht et al. (2013)).

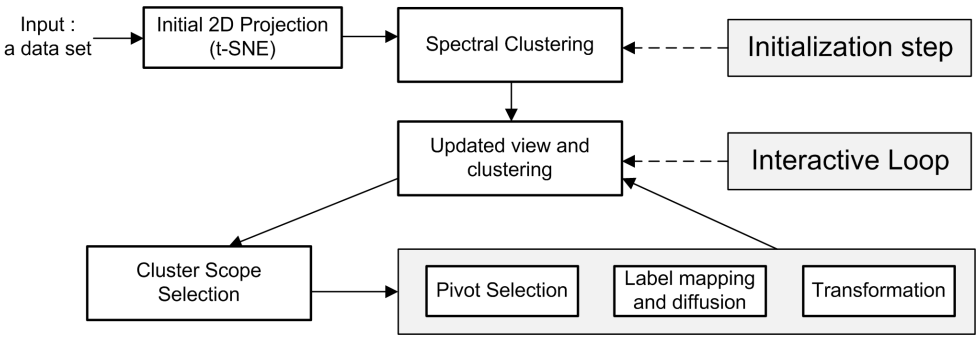


FIG. 1 – *Diagramme résumant le système proposé.*

Le clustering spectral est particulièrement adapté aux formes de clusters arbitraires alors typiquement obtenues. Une structure de clustering initiale est ainsi automatiquement estimée grâce à la méthode proposée dans Bruneau et al. (2014).

Cette projection, et la structure de clustering associée, sont utilisées comme initialisation du système décrit dans la figure 1. Des opérateurs de mise à jour sont alors mis à disposition de l'utilisateur, suivant des principes intuitifs (e.g. propager un label depuis un élément pivot, séparer ou regrouper deux composantes). Notre méthode se distingue en ne se basant que sur des propagations de labels et des transformations de la matrice de similarité décrivant les données traitées, sans requérir de ré-exécution de l'algorithme de clustering initial. La méthode t-SNE supportant des modifications incrémentales, la visualisation est adaptée après chaque opération de l'utilisateur. La figure 2 résume cette cinématique.

Un proof-of-concept de notre système a été testé sur les données COIL-20 (Nene et al. (1996)) et MNIST (LeCun et al. (1998)). Les résultats sont encourageants, mais nous avons identifié des limites à notre méthode, en lien avec le fonctionnement même de t-SNE. Des travaux sont en cours pour corriger ces problèmes, et développer un prototype plus abouti en termes d'interactivité.

Références

Andrienko, G., N. Andrienko, S. Rinzivillo, M. Nanni, D. Pedreschi, et F. Gianotti (2009). Interactive visual clustering of large collections of trajectories. *IEEE Symposium on Visual Analytics Science and Technology*, 3–10.

Bruneau, P., O. Parisot, et P. Pinheiro (2014). Une heuristique pour le paramétrage automatique de l'algorithme de clustering spectral. *EGC*.

Gisbrecht, A., B. Hammer, B. Mokbel, et A. Sczyrba (2013). Nonlinear dimensionality reduction for cluster identification in metagenomic samples. *17th International Conference on Information Visualisation (IV)*, 174–179.

LeCun, Y., L. Bottou, Y. Bengio, et P. Haffner (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE 86*(11), 2278–2324.

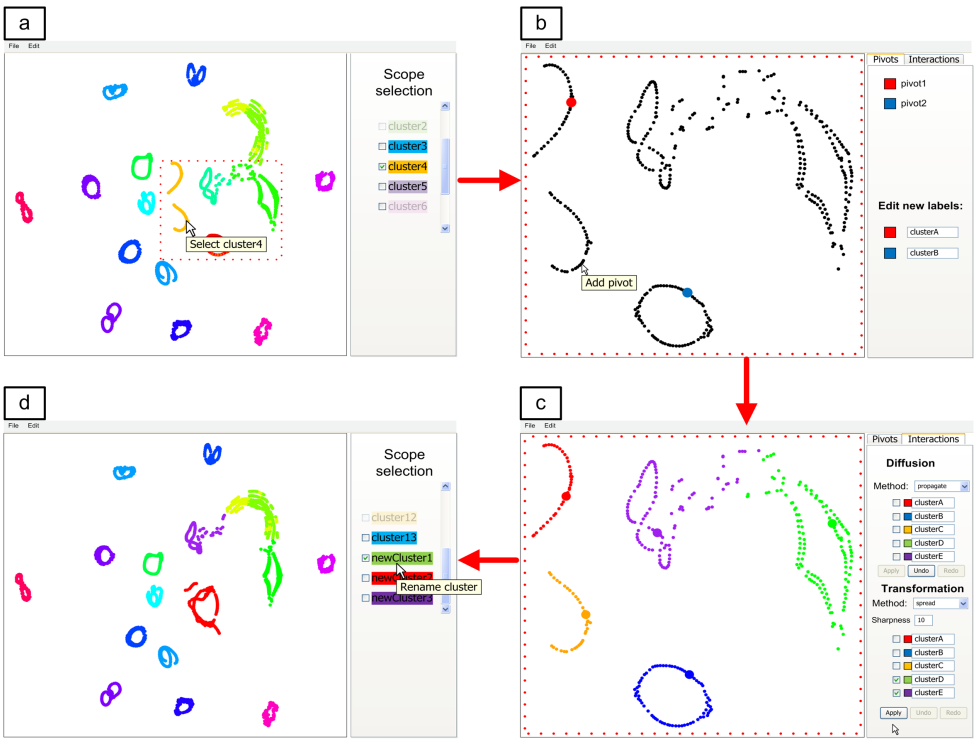


FIG. 2 – *Aperçu de la cinématique de notre système :* a) *Sélection d'une fenêtre d'interaction depuis une vue générale.* b) *Sélection de pivots d'interaction.* c) *Diffusion de labels depuis les pivots. Des opérations de séparation ou de regroupement de composantes peuvent être effectuées à partir des diffusions précédentes* d) *L'utilisateur peut ensuite fermer la fenêtre d'interaction.*

Lee, H., J. Kihm, J. Choo, J. Stasko, et H. Park (2012). iVisClustering : An interactive visual document clustering via topic modeling. In *Computer Graphics Forum*, Volume 31, pp. 1155–1164. Wiley Online Library.

Nene, S. A., S. K. Nayar, et H. Murase (1996). Columbia object image library (coil-20). Technical report, Technical Report CUCS-005-96.

Philippeau, J., J. Pinquier, P. Joly, et J. Carrive (2008). Dynamic organization of audiovisual database using a user-defined similarity measure based on low-level features. *IEEE International Conference on Image Processing*, 33–36.

Rinzivillo, S., D. Pedreschi, M. Nanni, F. Giannotti, N. Andrienko, et G. Andrienko (2008). Visually driven analysis of movement data by progressive clustering. *Information Visualization 7*(3-4), 225–239.

Seo, J. et B. Shneiderman (2002). Interactively exploring hierarchical clustering results [gene identification]. *Computer 35*(7), 80–86.

Turkay, C., J. Parulek, N. Reuter, et H. Hauser (2011). Integrating cluster formation and cluster evaluation in interactive visual analysis. *Proceedings of Spring Conference on Computer Graphics*.

van der Maaten, L. et G. Hinton (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research 9*, 2579–2605.

Summary

This article briefly describes a work in progress on an interactive visual clustering system. From an initial 2D projection, and an associated clustering, we contribute means to iteratively and interactively amend this initial representation. The user can thus inject his knowledge directly in the visualization, or visually fix potential errors of the clustering algorithm. The technique can also be used to help building a mental map of the processed data.

Index

B

Benoît Otjacques 45
Bertjan Broeksema 45
Bin Yang 1

F

Fatma Bouali 20, 29

G

Gilles Venturini 20, 29
Guillaume Santini 3

H

Hanane Azzag 3

J

Jean-Gabriel Ganascia 1

M

Mustapha Lebbah 3

N

Nhat-Quang Doan 3

O

Olivier Parisot 33

P

Philippe Pinheiro 33, 45
Pierrick Bruneau 45

T

Thomas Tamisier 33
Tianyang Liu 20, 29

Y

Yoann Didry 33

EGC 2014

Organisateurs

Sponsors