Hierarchical Visual Filtering, pragmatic and epistemic actions for database visualization

Jose F Rodrigues Jr
Institute of Mathematics and Computer Science
University of São Paulo
São Carlos, SP, Brazil
junio@icmc.usp.br

Carlos E Cirilo, Antonio F Prado
Computing Department
Fed. University of São Carlos
São Carlos, SP, Brazil
carlos_cirilo@dc.ufscar.br, prado@dc.ufscar.br

Luciana A M Zaina
Computing Department
Fed. University of São Carlos at Sorocaba
Sorocaba, SP, Brazil
lzaina@ufscar.br

ABSTRACT

Visualization techniques of all sorts suffer from visual cluttering, the occlusion of visual information due to the overlap of graphical items; and from excessive complexity in analytical tasks due to multiple parallel perspectives. To cope with these problems, we introduce Hierarchical Visual Filtering, a novel interaction principle based on pragmatic and epistemic actions. Pragmatic actions here mean that the analyst is able to visually select and filter information, determining visual configurations that reveal different perspectives; epistemic actions mean that the analyst can record, annotate, and recall intermediate visualizations created pragmatically. To do so, we use a tree-like organization to keep multiple visualization workspaces linked according to the analytical decisions took by the user. Our goal is to promote an innovative systematization that can augment the potential for database visual inspection, and for visualization systems in general. It is our contention that Hierarchical Visual Filtering can inspire a novel scheme of visualization environments in which space limitations and complexity are treated by means of interactive tasks.

Keywords

Information Visualization; Multiple Views; Visual Data Analysis; Databases; Interactive Filtering; Hierarchical Filtering

1 Introduction

The growth in information production has addressed the problem of accessing the riches of information embedded in large and complex databases. This issue has aggravated in yearly basis and one of the strategies to deal with it is to use visualization. However, sole visualization techniques are naturally limited in space. Users cannot tell apart items nor distinguish regions of interest when visualizations exceed the available display space, or when data dimensionality prevents effective presentations. Even using well-known interaction mechanisms such as interactive filtering [16], hierarchical parallel coordinates [6], focus+context [1], and link & brush [3], the grasping of interesting facts becomes a memory intensive task. A problem even worse when more than one investigation task is being performed over the same scene; in this situation, it is up to the analyst to remember the appropriate interaction parameters for each line of investigation and to switch in between them. These restrictions in data presentation lead to the need of interface and interaction designs that can aid the management of cognitive load in visual decision-support.

SAC'13 March 18-22, 2013, Coimbra, Portugal.

Cognitive aspects

In cognitive science, Kirsh and Maglio [11] identified two kinds of actions performed during decision-support activities. Pragmatic actions are performed to bring one closer to a goal, and epistemic actions are performed to uncover information that is hard to compute mentally. In arithmetic, for example [9], pragmatic actions permit to gradually advance on a problem's state in order to reach its solution. Epistemic actions correspond to various intermediate results (paper notes), which theoretically could be stored in working memory, but that are recorded externally to reduce cognitive loads. In the realm of Information Visualization (InfoVis) applications, pragmatic actions can be identified as interaction operations to lead the analyst in the task of discovering useful knowledge. Epistemic actions correspond to the recording of intermediate visual presentations in order to assist the analyst in a sequence of interactive steps. More precisely, in what refers to computational aided visualization, Kirsh and Maglio state that epistemic actions correspond to an automatic views management system whose advantages include:

1. reduced memory involved in visual analysis – space complexity;
2. reduced number of steps involved in visual analysis – time complexity;
3. reduced probability of error of visual analysis – reliability.

In a level higher than cognitive science, these advantages have been perceived by Grinstein and Ward [8] who enunciated that limitations in screen resolution and color perception can be solved through multiple linked visualizations. One main approach is to use multiple views enabled with focus+context functionalities to partition and detail intricate visualizations. Indeed, partitioning a dataset is critical to select relevant data and to reduce data for further investigation. Dividing the scene into multiple views help to create smaller subsets easier to be managed and that allow comparative discernment over different perspectives. Chi et at [2] state that a single complex view can be cognitively overwhelming. Multiple views can help the user to "divide and conquer" aiding memory by reducing the amount of data one needs to consider at the same time.

Visualizing relational databases
In the last decade, many works have addressed the issue of visualizing relational databases to formulate new hypotheses, to validate known hypotheses, or to simply present facts more intuitively. These works include multivariate multiple-view systems, and relational database driven systems.

Among the multivariate multiple-view systems, the GGobi system [4], and the Xmdv tool [17] employ linked views as defined by the Link & Brush principle. Our work differs from these former proposals by means of the hierarchical visual filtering principle. For this reason, our work cannot be directly compared to these other proposals, due to its innovative interaction scheme, and to its database orientation.

More specific to our work, in the field of relational-database driven systems, Wang *et al.* [20] present the ZoomTree, a web-based system that uses a grid layout to present sequential zoom operations carried over a database. Different from our approach, the ZoomTree allows data propagation via selection of dimensions to draw aggregation datacubes; also, the selection of data does not follow free database querying, instead, it is restricted to datacube-refinement operations. A pioneer work on the same topic is system Polaris [18], another grid-based layout whose main feature is its flexibility in defining the visual encoding; different from our work, Polaris does not allow hierarchical exploration, rather it improves on the commercial Pivot Table method. Another related proposal is the work of Mansmann and Scholl [12]; their Decomposition Tree permits the user to visually conduct datacube operations, that is, dimension-oriented operations that are data-driven, rather than analysis-driven. As a last remark, we can state that our work is defined over a combination of visualization with clutter reduction techniques; in this scenario, even if we consider the taxonomic review of clutter-reduction techniques of Ellis and Dix [5], we find that our interaction scheme is a novel contribution.

Visual filtering
Visual filtering (or brushing) over multiple coordinated views [15] is one of the main interaction techniques for visualization; it allows analysts to select the graphical items that are more interesting, gaining focus and details over them. In a recent work, Weaver [22] introduces a scheme that is opposite to ours. Instead of creating multiple views by means of visual filtering, they define a collective filtering based on selections originating from multiple views. Kehrer *et al.* [10] argue in favor of statistical summarizations (mean, variance, skewness, and kurtosis), each in a dedicated view, as a means for interpreting visually filtered data. Also recent, Turkay *et al.* [19] create multiple views by simultaneously filtering data and data dimensions assisted by statistical summaries. In another work, Weaver [21] proposes the use of boolean logic to progressively refine the content visually presented, leading to a more robust filtering. Our work differs from all these proposals as it permits an unrestricted number of views that are kept linked by complimentary actions; besides, our proposal is adequate for any kind of visualization technique, including statistical summaries as well.

The rest of the text is organized as follows. In section 2 we present the innovations proposed by our system along with concepts, analyses, and discussion. In section 3 we formally analyze the potential of our interaction principle regarding visual clutter. In section 4 we demonstrate our technique in respect to pragmatic and epistemic actions, just before the final remarks of section 5.

2 Methodology
As a critical setting for database visual analysis, we consider the use of heterogeneous visualizations in a multiple views environment making use of pragmatic and epistemic actions. In order to prove this concept, we developed Visualization Tree (VisTree) [13], a system that enables multiple representations of a dataset relation allowing comparison and operation with a strong feeling of locus of control. VisTree is designed for any kind of visualization technique; specifically for this work, we use classical techniques Parallel Coordinates, Scatter Plots, Table Lens [14], and Fastmap-based projection [7]. Over the VisTree systematization we demonstrate the Hierarchical Visual Filtering principle.

2.1 Hierarchical Visual Filtering
A straightforward way to partition a dataset is to use relational database queries that, over visualization scenes, correspond to interactive filtering, a well-known pragmatic instrument of visualization. Extending this interactive principle, here we propose what we call *Hierarchical Visual Filtering*, an improvement of interactive filtering that brings epistemic possibilities to visual analysis. Hierarchical Visual Filtering allows for pragmatic actions as the analyst is able to select parts of the visualization scene; and allows for epistemic actions as the analyst is able to record, annotate, and recall intermediate visualizations created over his pragmatic actions. Hierarchical Visual Filtering contributes to visual analysis research in two ways:

- by reducing visual clutter for more scalable analysis;
- by reducing cognitive loads – specially over memory, for a richer analytical experience.

Hierarchical Visual Filtering combines interactive selection and progressive refinement to permit analytical management in a hierarchically-arranged environment. As presented in Figure 1, it is composed of a dataset D, a visualization V, a visualization function v, an interactive filtering function f and a function Λ that plays just reverse to function v.

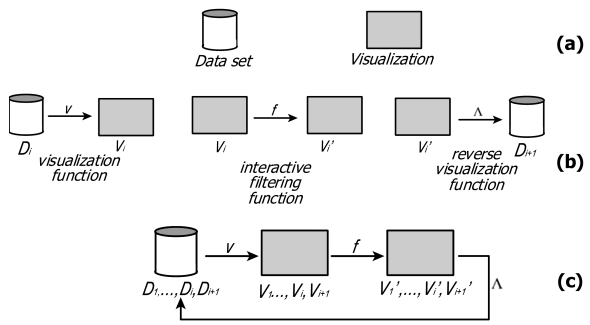

Figure 1: (a) Components of Hierarchical Visual Filtering, (b) functions that define it, and (c) its iterative cycle.

In Figure 1, function $v : D_i \rightarrow V_i$ is parameterized by a pair (s, g) where s is the spatialization scheme that states how data D occupy the screen space (projection, graph-like, sequentially, and so on) and g is the set of graphical marks (dots, lines, curves, icons, and so on) used by visualization V_i. Also in Figure 1, filtering function $f : V_i \rightarrow V_i'$ is parameterized by a pair (d, e) where d is the set of dimensions of the data and e is a set of *relational select predicates* that apply to d in order to determine the selection of the interactive filtering. Interactive filtering produces an altered configuration V_i' that is made of a subset of the graphical entities of visualization

V_i. Function $\Lambda : V_i' \rightarrow D_{i+1}$ receives a filtered visualization V_i' and performs the opposite of function v using its same parameters; it returns the data items D_{i+1} that define visualization V_i'.

Figure 1 presents the iteration cycle that defines Hierarchical Visual Filtering. According to it, a dataset D_i is used to create a visualization V_i through function v. This visualization is interactively filtered via function f to determine a new visualization configuration V_i'. Then, the data that is being presented in V_i' is extracted with function Λ. With the extracted data D_{i+1}, a new visualization V_{i+1} can be created. This process is to be repeated iteratively according to the user's exploratory goals, which determines how many steps are necessary.

In Figure 2, we illustrate the Hierarchical Visual Filtering the way it is performed in VisTree and according to the technique presented in Figure 1. We use two iteration sequences, in the first one the entire dataset ("SELECT * FROM DATA_SET") D_1 is loaded in a new visualization workspace; the interface of this workspace permits to choose one of five available visualization techniques – Figure 2(a). By choosing one of them, visualization function v is triggered to generate visualization V_1. Over V_1, the user can determine parameters (highlighted in the figure) for interactive function f, and a new visualization V_1' is determined. Another user command (Pipeline – Figure 2(b)) triggers function Λ, which identifies the data D_2 being presented in V_1' ("SELECT * FROM DATA_SET WHERE ((139 $\leq$ DIM_2 $\leq$ 334) AND (73 $\leq$ DIM_4 $\leq$ 78))"). The data of V_1' is now sent (pipelined) to a new workspace to create visualizations V_2 and then V_2' in the second iteration cycle – Figure 2(c).

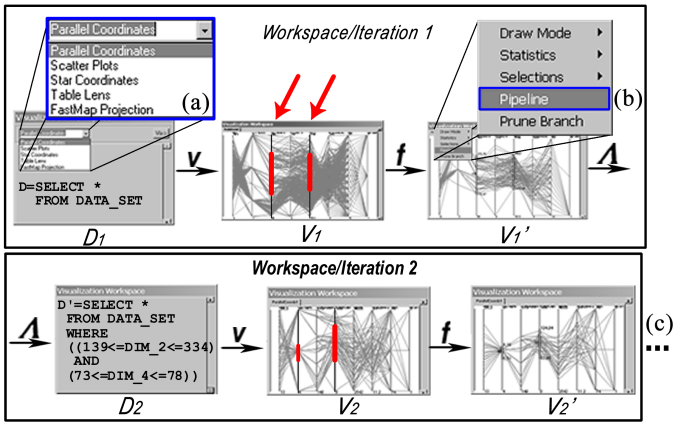

Figure 2: Hierarchical Visual Filtering iteration loop. (a) Choice for visualization function v; (b) triggering of function Λ; (c) second iteration cycle.

Hierarchical Visual Filtering permits the user to define new visualization workspaces, each one carrying a subset of the data that he/she considers worthy for analysis. To do this, the user can observe a given workspace and interactively filter its data by choosing a set of parameters e of relational select predicates. When an interesting visualization configuration comes up, the user can command its data elements to flow to a new workspace. This process can be repeated for the new workspace or for the same workspace. As another workspace is created, the current branch of exploration becomes deeper, as seen in Figure 3(a). If a new workspace is created from the same workspace, a parallel branch of exploration is created, as seen in Figure 3(b). The same operation can be done at any level of the tree, as demonstrated in Figure 3(c). Along this iteration/interaction, the VisTree systematization is responsible for keeping track of the multiple workspaces and to present them in a tree-like structure.

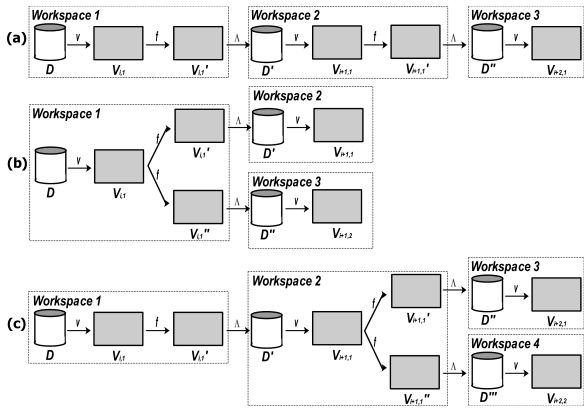

Figure 3: VisTree Construction. (a) Sequential exploration of a branch. (b) Parallel exploration over different branches. (c) Iteration at deeper levels of the tree.

The epistemic aspect of VisTree comes as the system automatically records the relational predicates used during analysis and, also, it comes as the user is able to annotate each workspace so that she/he can remember what the presented data refers to. VisTree complements these features by organizing the workspaces in accordance to how they were created, providing a sense of analytical flow with reduced cognitive load.

2.2 Tree management and interaction

As the user pragmatically creates new workspaces and the tree structure evolves, it is necessary to manage its structure in order to maximize space utilization and to maintain the epistemic potential of the tree representation. Each node of the tree occupies an equal area of the 2D space, therefore it is necessary to partition this space at the same time that the tree representation adjusts to its spatial arrangement.

We developed a recursive algorithm to combine suitable tree presentation with space partitioning of display space. The algorithm uses a tree scheme named *tNode* – presented in Figure 4(a), which carries pointers to son and sibling nodes. It also carries a pointer to an instance of a visualization object, referenced as *vis* pointer. The tree data structure grows as the user triggers interaction events, as illustrated in Figures 4(b) and 4(c). The structure is used during the whole management of the visualization environment.

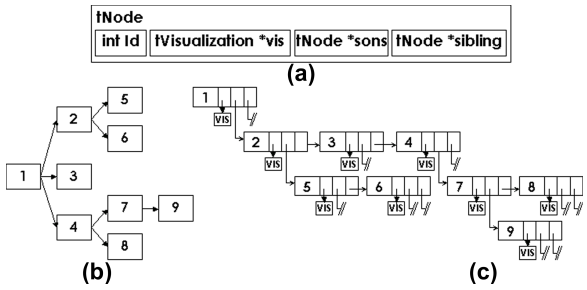

Figure 4: (a) The node structure for managing the VisTree. (b) A tree example and its correspondent data structure. (c) The *vis* pointers hold the instances of the visualizations (workspaces) being presented.

In VisTree, the tree is presented in left to right orientation, instead of top-down as usual. To organize nodes this way, we break the node positioning problem in two parts; first, to determine horizontal positioning and, then, to determine vertical positioning. For a given node, horizontal positioning is straightly achieved using the node's (horizontal) level, which corresponds to how deep the node is located in the tree, as can be seen in Figure 5. Meanwhile, vertical positioning is a little trickier because it demands positioning from the bottom of the display up to its top considering the number of workspaces and of branches at each level. Therefore, tree presentation must start by the leaves, otherwise we may have edge crossings and/or nodes overlapping that could compromise the tree aesthetics.

In Figure 5 we illustrate how the presentation process occurs. The first thing is that, any new node is positioned at the bottom of the screen. In steps 1 through 3 in Figure 5, nodes 5 and 6 illustrate the case for nodes without sons: initial bottom positioning, search for the first node above it (surrounded by circle) and then repositioning. In steps 4 through 8, nodes 7 and 8 illustrate the case for nodes with sons; after the first repositioning, it is necessary to perform a second repositioning according to the child nodes (surrounded by an ellipse). For aesthetic reasons, this second repositioning places the father node in a position where both the sum and the variance of distances for all child nodes are minimized. That is, the father node is positioned in the middle height, which ranges from its highest son (or grandson) to its lowest son (or grandson).

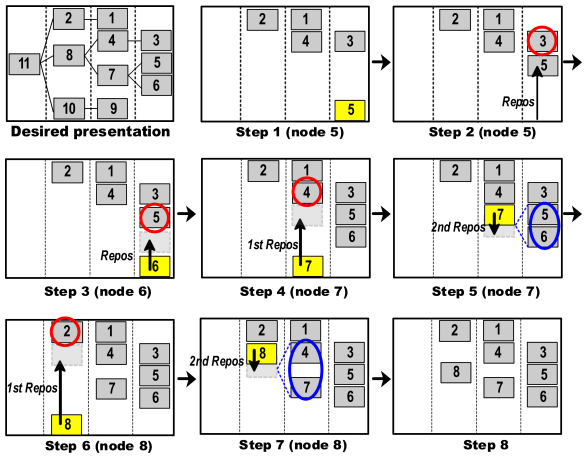

Figure 5: VisTree's nodes positioning. The first illustration presents the desired presentation. Following, steps 1 through 8 illustrate the positioning of nodes 5 through 8 in order to achieve the desired configuration. For each positioned node, its "first node above" is highlighted with a circle. Child sons that are considered for repositioning are highlighted with ellipses.

2.3 Workspaces management and interaction

In the environment of the VisTree, as new epistemic information is recorded in the form of new workspaces, the tree grows and its nodes become smaller in terms of screen space. However, the nodes of the tree have a minimum acceptable size, otherwise, too small visualization scenes could not be observed. For this minimum size, we use 2.5 x 1.5 inches, which nearly corresponds to a two-column article illustration. We assume that the screen resolution will always be sufficient to support such presentation size, no matter the technique. This minimum size implicates that, at a certain step, the algorithm can no longer accommodate the tree inside the display space, because the nodes would have to satisfy spatial constraints causing them to be smaller than what is practically visible.

Our strategy to handle this matter is to provide automatic and manual interaction mechanisms to explore the visualization environment. Hence, when the tree does no longer fit the screen space and a new node is created, we automatically position this last node at the center of the screen. With this mechanism we provide as much context as possible for the new node; at the same time, we induce locus of control, because the last user operation is immediately presented to him/her surrounded by the formerly created workspaces.

But, as we provide endless visualization space, the drawback is that the tree can be much bigger than the screen space and acting epistemically becomes a usability problem. To tackle with this, we designed an interaction scheme enabled with zooming and translation functionalities. These two features aid in the process but, for large trees, they may demand excessive interaction steps in order for the user to grasp the desired information. Therefore, we also designed a novel interaction scheme with two features, named *Instant Focus* and *Instant Context*:

- Instant Context, Figure 6 (a), works by presenting the entire tree structure no matter how small its nodes become. To do so, the user has to pass the mouse over a desired node and the tree will be visualized with emphasis over that node. It becomes possible to figure out where in the tree structure the node is located. As the user moves the cursor out of the desired node, the application returns to the former spatial arrangement;
- Instant Focus, Figure 6(c), allows the user to pop up a separate window from the tree structure; this new window bears the visualization workspace that the user has chosen for focusing. To do so, similarly, the user must pass the mouse cursor over the desired workspace. The focus window can occupy part of the screen space or can be maximized for the entire screen. To return to the VisTree environment, the user has to pass the mouse cursor out of the boundaries of the focus window or to close the window in case it has been maximized.

Instant Focus and Instant Context permits the user to benefit from focus+context interaction over the VisTree multiple windows environment.

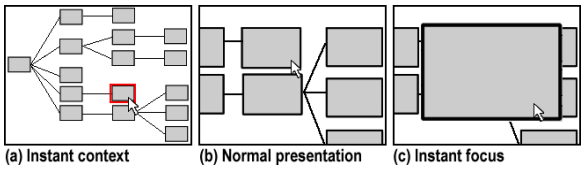

Figure 6: Instant Context shows where in the tree structure a node of interest is located (a). For large trees, normal presentation may hide context and/or detail (b). Instant Focus presents the details of a given scene by increasing its window size (c).

3 Reduction of visual clutter

In this section we formally demonstrate how the refinement promoted by Hierarchical Visual Filtering deals with overlap of graphical items reducing the problems of visual clutter.

Problem formalization

Initially, we formalize the concepts of *graphical item*, *screen-space*, and *screen-space coordinates*. We consider that *graphical*

items refer to any distinct visual mark used for data visualization, *screen-space* refers to the available display space that a given visualization can use for presentation, and *screen-space coordinates*, in turn, refer to each position that can be addressed in a given screen-space.

The screen-space is naturally discrete in the number of coordinates, no matter whether the domain of the data being presented is discrete or continuous. Consequently, the number of coordinates in screen-space limits the number of graphical items that can be presented, what varies depending on the data domain that will be presented.

Although the number of possible graphical items can be much bigger than the number of screen-coordinates, the number of possible graphical items cannot exceed the number of screen-coordinates. This problem is even worse for graphical items that are bigger than one pixel. This limitation determines that visualization techniques suffer from overlap of graphical items and, consequently, by visual cluttering. In this context, overlap of graphical items is understood as the mapping of different data items to the same screen coordinate; meanwhile, *visual cluttering* refers to the confused or disordered arrangement that arises from overlap of graphical items, what negatively affects perception.

Overlap of graphical items

For this topic, we initially formalize the concept of *region of interest*. For a given visualization technique, a region of interest is a sub-area of the screen-space in which a collection of data items is being presented. The biggest region of interest is the whole visualization itself, meanwhile, smaller regions of interest are defined via interactive filtering.

Based on these notions, we can develop a formal metric for the overlapping of graphical items. The overlap can be measured by the density of elements that map to one same visual coordinate. Therefore, given a workspace Δ composed by the collection D of data items, and by the set G of graphical items, the overlap factor is given by the total number of elements mapped to Δ, that is, $|D|$, divided by the total number of graphical items presented in Δ, that is, $|G|$. Then:

$$overlap\_factor = \frac{|\text{data\_elements\_mapped\_to\_}\Delta|}{|\text{graphical\_items\_presented\_in\_}\Delta|} = \frac{|D|}{|G|} \quad (1)$$

The same holds for regions of interest. That is, given a region of interest $\delta \subset \Delta$, the $overlap\_factor$ is given by the total number of elements mapped to this region, that is, $|d|$ for $d \subset D$, divided by the total number of graphical items presented in this region, that is, g for $g \subset G$.

As presented in Figure 7, the Hierarchical Visual Filtering determines that, given a region of interest $\delta \subset \Delta$ with $overlap\_factor = |d|/|g|$, it is possible to create a new workspace Δ' with $overlap\_factor' = |d|/|G'|$. Considering that a region of interest is always smaller or equal to the screen-space, and considering that the workspaces created via pipelining use the whole screen-space, then, the following inequality is always valid $|g| \leq |G'|$. This is because a wider screen-space will be available for the new pipelined workspace and, thus, more graphical items can be presented. Consequently, $overlap\_factor' \leq overlap\_factor$ is always true. In other words, the pipelining operation will always create new scenes with equal or reduced $overlap\_factor$.

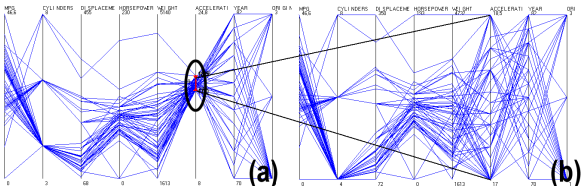

Figure 7: Treatment of graphical items overlap via pipeline refinement. (a) A workspace Δ and a region of interest $\delta \subset \Delta$ (black ellipse) defined with parameters $17.0 \leq ATTRIBUTE6 \leq 18.5$. (b) The workspace Δ' achieved via pipelining of the region of interest presented in (a). Also in (b), it is clear the increase in the number of graphical items, specially for $ATTRIBUTE6$. As consequence, in most dimensions not only the overlap factor was reduced but also the visual clutter in general.

4 Experiments over pragmatic and epistemic actions

In this section we demonstrate the possibilities of the Hierarchical Visual Filtering; to do so, we perform experiments over a dataset of agrometeorological data - the Embrapa dataset. The dataset has 9 attributes: precipitation, maximum temperature, minimum temperature, normalized difference vegetation index (NDVI), water requirement satisfaction index (WRSI), average temperature, potential evapotranspiration (ETP), real evapotranspiration (ETR) and measured evapotranspiration (ETM) collected partly with remote sensors (satellite) and partly with *in locus* samples from sugar cane plantation regions in Brazil. The data was collected during 82 months from 2001 to 2007, so that each record corresponds to the data collected in a given month for one region. Since there are 5 regions: Araraquara (1), Araras (2), Jaboticabal (3), Jaú (4), and Ribeirão Preto (5), there is a total of 410 records. In order to test our hypothesis, we proceeded with the following tasks:

- characterize the 5 regions in relation to all the attributes of the dataset;
- choose two regions of interest in order to draw further visual clues comparatively.

The problem here is that the data is organized according to several regions and, although we want to inspect these regions separately, we do not want to use multiple visualization sessions. Multiple sessions would demand managing several windows, or even paper annotations. Also, the data is multivariate and it would be interesting to observe it according to multiple kinds of visualization techniques put together in a single environment. Lastly, it is desired to select subsets of the data and to summarize them by means of statistical calculations drawn over each of the visual workspaces, making the comparative analysis faster. In Figure 8, we show a visualization session for these data and for the proposed tasks; the image was edited for better presentation.

The visualization of all the data items over Parallel Coordinates – Figure 8(a) – presents a hard to interpret cluttering of lines. For this, the first step of the analytical process was to define 5 filterings each corresponding to a different region. With the filterings we could create 5 workspaces – Figure 8(b) – presented simultaneously and prone to detailed inspection by means of Instant Focus. Next, we want to characterize the workspaces' data with a simple statistical operation - to do so, we calculate the average of each dimension at each workspace and proceed by drawing a polyline of average values emphasized in green on top of the Parallel Coordinates – also presented in Figure 8(b). This first round shows that the 5

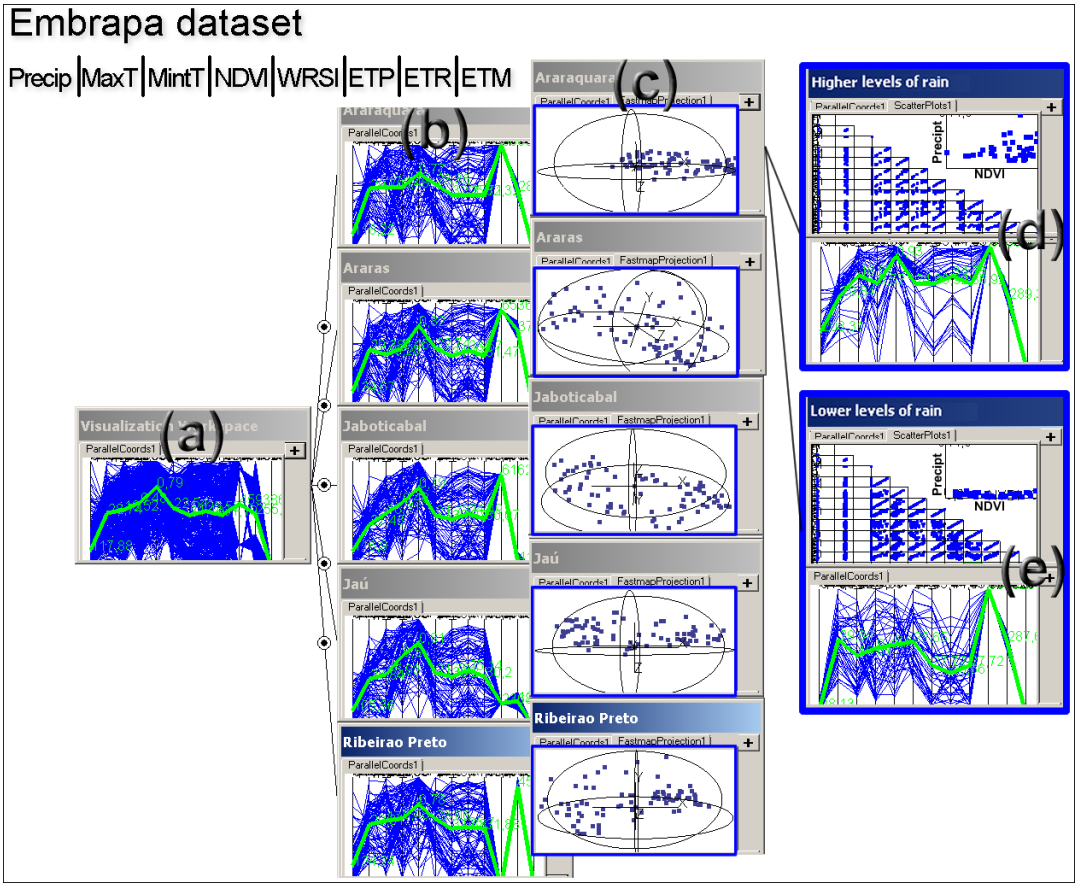

Figure 8: Definition of an analytical hierarchy of data over the Embrapa dataset. (a) Visualization of all the data items. (b) Interactive filtering of 5 subsets of the data, with one workspace to each showing a Parallel Coordinates scene. (c) The 5 workspaces, now visualized with Fastmap multidimensional projection. (d) Workspace with the 50% records with highest levels of precipitation of region 1. (e) Workspace with the 50% records with lowest levels of precipitation of region 1. Each workspace carries details of its correspondent select predicate, number of records, and user annotations (header of each window).

regions have similar values for each of the attributes with slight variations that confer them local peaks according to each average polyline.

In a next step, we want to visualize the same data in a multidimensional projection achieved using the well-known Fastmap algorithm; as so, in Figure 8(c), another visualization is presented for each of the workspaces. These new visualizations work by mapping the 9 attributes of the datasets into 3 dimensions, allowing one to inspect the data in respect to their dispersion in space, to the presence of clusters, and of outliers. The figure shows that region 1 (upper-most workspace) is the one whose data points better characterize a single cluster, contrasting with the others that define widely spread spatial regions. From this observation we assume that this region should be better analyzed, since evidence indicates that its data is homogeneous, potentially following a well-defined pattern. On the other hand, region 2 presents the plotting in which the points are more widely spread if compared to all the other workspaces, indicating a high-level of irregular behavior, what should be further investigated as well.

Next, we visually filter region 1 according to the 50-th percentile of attribute precipitation. This way we get a level deeper in the hierarchy and two new workspaces that we visualize according to techniques Parallel Coordinates and Scatter Plots. The first visualization accounts for the records of region 1 that present higher levels of rain – Figure 8(d), while the second accounts for the records with lower levels of rain – Figure 8(e). By inspecting the Parallel Coordinates average summary of both workspaces, there is a clear distinction in all of the attributes when the level of rain significantly varies. More specifically, the Scatter Plots visualization shows that the NDVI x Precipitation plot indicates that only part of the vegetation suffers with rain shortage, while a great part of it remains with high levels of green matter, possibly due to artificial irrigation. The same findings were verified for region 2, although not presented in the figure due to space limitations.

The demonstration of our system is not favored by the flat paper presentation; for this reason we put the system fully operational at `http://gbdi.icmc.usp.br/~junio/VisTree/VisTree.htm`, where it can be experimented with several other datasets.

User tests

In order to test the Hierarchical Visual Filtering in terms of usability, the same analytical tasks as described previously in this section were proposed to a group of 22 Computational Physics undergraduate students, familiar with computational tools. The students were instructed to use the VisTree system and to perform the same tasks, but in two different ways: in a first round, using Hierarchical Visual Filtering; in a second round, using one single workspace with simple visual filtering in order to create all the visualizations.

Under supervision, the students were monitored so that wall-clock time could be measured, and so that it would be possible to verify whether the tasks were really executed – without shortcuts. The experiment was conducted with the students, one at a time, in the same machinery, according to the following protocol:

- first, the students were trained with another dataset in order to get familiar with the system;
- the students received the expected answer of the analytical experiment: identify the two extreme regions, and to create further visualizations from each based on the precipitation attribute; this way, it would be possible to analyze only the Hierarchical Visual Filtering, and not the entire system;
- in the second round, the students received a different version of the dataset in which the labels of the regions differed from the dataset used in the first round; this way they would still have the challenge of identifying the two regions of interest;
- for the second round, the students could use paper to make annotations;
- for each task completed, the visualization was verified and the elapsed time was annotated.

In this experiment, 21 of the students were able to suitably complete the exercise. From these, all of them could complete the task faster by using the Hierarchical Visual Filtering, the gain was of 42% average – 4 minutes 52 seconds average for the first round, and 8 minutes 24 seconds average for the second round. Only 5 students used the paper for annotation, the remaining subjects preferred to use multiple windows and to alternate in between them. All of the subjects declared that the tasks were easier to be accomplished if Hierarchical Visual Filtering could be used.

For these results, we argue that the time gains would be higher for more complex tasks, or even for different tasks, instead of the same tasks performed with different operations; the same applies for the need of paper annotations. In despite of that, the results confirmed the adequacy and potential of the Hierarchical Visual Filtering analytically and empirically.

5 Conclusions

We have introduced Hierarchical Visual Filtering, an innovative interaction principle that overcomes cluttering issues and analytical complexity. Its principle is to allow the user to pipeline subsets of information to multiple workspaces that are automatically managed over a tree-like presentation. The essence of our contribution is based on actions that are pragmatic – filter and pipeline, and epistemic – record, annotate, and recall persistent visualizations. We have performed experiments that demonstrate that our systematization potentially speeds up and simplifies analytical tasks.

For now, we have experienced Hierarchical Visual Filtering over one single table (database relation); as further goals, we envision the design of VisTree as a visual join tool, in which multiple related tables could be inspected. This design shall bring new challenges as join information may demand massive memory and processing resources; graphically, it will be necessary to adapt the Hierarchical Visual Filtering to a graph-like presentation, since a tree will not be enough; in terms of interaction, such configurations may be tricky as tables can be interrelated according to different cardinalities.

6 Acknowledgments

This work was partly supported by CNPq (Brazilian National Research Foundation), CAPES (Brazilian Committee for Graduate Studies), Microsoft Research and FAPESP (São Paulo State Research Foundation).

7 References

[1] G. Bisson and R. Blanch. Improving visualization of large hierarchical clustering. In *Information Visualisation*, pages 220–228, 2012.

[2] E. H. Chi, J. A. Konstan, P. Barry, and J. Riedl. A spreadsheet approach to information visualization. In *ACM User Interface Software and Technology*, pages 79–80, 1997.

[3] J. Claessen and J. van Wijk. Flexible linked axes for multivariate data visualization. *IEEE TVCG*, 17(12):2310–2316, 2011.

[4] D. Cook and D. F. Swayne. *Interactive and Dynamic Graphics for Data Analysis With R and GGobi*. Springer, 2007.

[5] G. Ellis and A. Dix. A taxonomy of clutter reduction for information visualisation. *IEEE TVCG*, 13(6):1216 –1223, 2007.

[6] Y.-H. F., M. Ward, and E. Rundensteiner. Structure-based brushes: a mechanism for navigating hierarchically organized data and information spaces. *IEEE TVCG*, 6(2):150–159, 2000.

[7] C. Faloutsos and K. Lin. Fastmap: a fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. *SIGMOD Rec.*, 24(2):163–174, 1995.

[8] G. G. Grinstein and M. O. Ward. Introduction to data visualization. *Information visualization in data mining and knowledge discovery*, pages 21–45, 2001.

[9] G. Hitch. The role of short-term working memory in mental arithmetic. *Cognitive Psychology*, 10:302–323, 1978.

[10] J. Kehrer, P. Filzmoser, and H. Hauser. Brushing moments in interactive visual analysis. *Comput. Graph. Forum*, 29(3):813–822, 2010.

[11] D. Kirsh and P. P. Maglio. On distinguishing epistemic from pragmatic action. *Cognitive Science*, 18(4):513–549, 1994.

[12] S. Mansmann and M. H. Scholl. Exploring olap aggregates with hierarchical visualization techniques. In *ACM SAC*, pages 1067–1073, 2007.

[13] J. F. Rodrigues Jr, A. J. M. Traina, and C. Traina. Visualization tree, multiple linked analytical decisions. In *Smart Graphics (LNCS 3638/2005)*, pages 65–76. Springer-Verlag, 2005.

[14] R. Rao and S. Card. The table lens: Merging graphical and symbolic representation in an interactive focus+context visualization for tabular information. In *Proc. Human Factors in Computing Systems*, pages 318–322, 1994.

[15] J. Roberts. State of the art: Coordinated multiple views in exploratory visualization. In *Coordinated and Multiple Views in Exploratory Visualization*, pages 61–71, 2007.

[16] J. Roberts and M. Wright. Towards ubiquitous brushing for information visualization. In *Information Visualization Conf.*, pages 151–156, 2006.

[17] E. A. Rundensteiner, M. O. Ward, Z. Xie, Q. Cui, C. V. Wad, D. Yang, and S. Huang. Xmdvtool^9: : quality-aware interactive data exploration. In *SIGMOD Conference*, pages 1109–1112, 2007.

[18] C. Stolte, D. Tang, and P. Hanrahan. Polaris: a system for query, analysis, and visualization of multidimensional relational databases. *IEEE TVCG*, 8(1):52 –65, 2002.

[19] C. Turkay, P. Filzmoser, and H. Hauser. Brushing dimensions - a dual visual analysis model for high-dimensional data. *IEEE TVCG*, 17(12):2591–2599, 2011.

[20] B. Wang, G. Chen, J. Bu, and Y. Yu. Zoomtree: Unrestricted zoom paths in multiscale visual analysis of relational databases. In *Computer Vision, Imaging and Computer Graphics*, volume 229, pages 299–317. 2011.

[21] C. Weaver. Conjunctive visual forms. *IEEE TVCG*, 15(6):929 –936, 2009.

[22] C. Weaver. Cross-filtered views for multidimensional visual analysis. *IEEE TVCG*, 16(2):192–204, 2010.