Eurographics Conference on Visualization (EuroVis) (2012)
M. Meyer and T. Weinkauf (Editors)

MCS Filters to Express Partial Satisfaction of Criteria

B. Otjacques[1], M. Cornil[1], M. Stefas[1], F. Feltz[1]

[1]Public Research Centre - Gabriel Lippmann, Luxembourg

Abstract

This paper describes a new graphical component called Multidimensional Concentric Sliders (MCS) to visually build Boolean conjunctive queries for numerical attributes. The main originality consists in allowing the user to impose a strict observance for some criteria as well as to offer options to be more tolerant for others. The component uses color coding to visualize groups of elements that approximately satisfy the query criteria to a similar degree (e.g. full satisfaction of all criteria, near-misses, not acceptable at all). The paper discusses both the concepts and a prototypal implementation.

Categories and Subject Descriptors (according to ACM CCS): H.3.3 [Information Storage And Retrieval]: Information Search and Retrieval—Query formulation

1. Introduction

Someone who wants to buy an apartment must find specific items satisfying some criteria in a whole dataset. Nowadays, visual components are largely used to specify the criteria (e.g. sliders) as well as to display the results (e.g. map).

More advanced solutions should also support the users to identify the elements that do not exactly satisfy all the criteria but may nevertheless interest them. For instance, someone looking for an apartment having at least two bedrooms and a floor area between $50m^2$ and $70m^2$ may be interested in a $49m^2$ apartment with two bedrooms even if it does not exactly match what he has encoded in the query engine. In this case, $50m^2$ is in fact an approximate value encoded as a strict limit due to the user interface design. On the other hand, a $60m^2$ apartment with only one bedroom may not be acceptable because the number of bedrooms is really a strict limit for him. In other terms, there is a need for querying systems helping the user to distinguish the elements just failing to satisfy one or several criteria from items that are definitely too far from the perfect answer (i.e. classic result set of the query). Furthermore, the user should also have the possibility to express how strictly a given criterion should be respected from his perspective.

The purpose of our research is to design visual queries going beyond searching for and visualizing the elements which strictly satisfy several criteria. The system should also show the results approaching what the user wants and provide information about the degree of query satisfaction.

2. State-of-the-Art

Research on visual queries can be grouped according to what is primarily under focus: (1) to express a question with visual means, (2) to show the results and the relationship to the question or (3) to combine both aspects very closely.

In the first category, the purpose is to help users to build queries by combining visual components. For instance, the visual interface Filter/Flow [YS93] helps the user to specify queries with Boolean operators (AND, OR and NOT) by using the metaphor of water flowing through filters. VQuery [Jon98] is another approach using Venn-like diagrams.

Second, the visualization of the filtered data has also been much investigated. VisDB [KK94] displays the elements that satisfy (or not) each criterion of a given query. Pixels representing database items are arranged and colored to provide information about the degree of satisfaction of the query. The list-based representation of web search results called HotMap [HY06] is another example. We can also mention the visual query language KMVQL [Huo08] based on Karnaugh maps.

In the third case, the data view and the query are fused in the user interface. For example, InfoCrystal [Spo93] proposes a view that presents data filtered by criteria. It uses

DOI: 10.2312/PE/EuroVisShort/EuroVisShort2012/049-053

several visual variables like shape, size, color or orientation to code information about the dataset and the query. The Attribute Explorer [ST98] dynamically links some interactive histograms (one per data attribute) to show how many criteria are satisfied by each element. Color coding is also used to let the user know which criteria should be modified for adding/removing some elements into the result before having to move any cursor (cf. concept of sensitivity information). Otjacques et al. [OCSF11] have proposed to map the query specification to a planar space divided into concentric zones showing the degree of satisfaction of the criteria. Unfortunately this approach is only valid for queries with two numerical criteria. TimeSearcher [HS04] is another example where queries are composed by drawing rectangles directly in the data view.

In all of these contexts, direct manipulation is acknowledged to add value providing that the refresh time be immediate, letting the user have a sense of causation [AWS92]. The dynamic HomeFinder is a seminal example [WS92] using some sliders for specifying criteria as ranges of values. The dynamic interface that instantaneously updates the data views helps the user to know if an element is far from or close to the query result: by moving a cursor he can see some elements appearing or disappearing from the selection. However, this information can not be captured with static snapshots (which may be a weakness in some circumstances).

To provide some direct or indirect means to understand the partial satisfaction of query criteria, these proposals use various visual cues combining spatial layout (e.g. [KK94]), color coding (e.g. [ST98]) and/or animated data views (e.g. [WS92]). The concept of "Smooth Brushing" [DH02] [HLD02] has a similar purpose: a DOI (Degree-Of-Interest) function is applied to specify the value of a given visual variable according to the position within focus or context regions of the elements to be displayed.

3. Multidimensional Concentric Sliders: Concepts

Problem Statement

We consider a dataset of elements E_i having n numerical attributes (i.e. categorical data is not considered here). The Boolean conjunctive query Q is composed of m criteria. Each criterion specifies an interval of acceptable values for a given numerical attribute of the elements E_i. The problem can be formally described as:

- $x_i = (x_{i,0}, x_{i,1}, ..., x_{i,n})$, vector of all attribute values of E_i
- $x_i^* = (x_{i,0}^*, x_{i,1}^*, ..., x_{i,m}^*)$, vector of attribute values of E_i that are involved in a query criterion
- Query $Q := \{E_i | a \leq x_i^* \leq b\}\, a, b, x_i^* \in \mathbb{R}^m$

This paper focuses on how to visually specify the query Q and how to show how much the elements E_i satisfy it. For instance, how to allow a customer to search for an apartment with a floor area between $50m^2$ and $70m^2$ and a number of rooms between 2 and 4 and to also take into account that he might also accept an apartment close to these limits? The customer may also have a different sensitivity to each criterion (e.g. rent and floor area are usually valued differently by a family and by a single). Furthermore, this sensitivity is often asymmetric (e.g. a rent below the lower limit will be more easily accepted than a rent higher than the upper limit). This aspect will lead us to introduce a metric expressing how much a given element E_i satisfy a query Q. For example, how far is a $55m^2$ apartment with only one room from the customer request previously mentioned?

Finally, we want to design a visual query system that may be used with various data views. Indeed, each visualization technique has its own strengths and weaknesses. The nature of the data as well as the type of investigation that the user wants to carry out determine the most appropriate view.

Query Layout

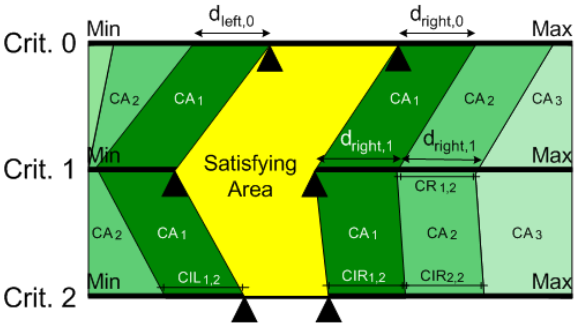

Figure 1: *Query Layout.*

Each query criterion is associated to a slider, which is a very usual strategy. The range of values of these sliders is mapped to the corresponding range of observed values in the data set. The originality of our proposal is to group all sliders into one component (cf. Figure 1) where they are placed one above the other. A criterion is specified as an interval of values via two cursors of the corresponding slider. Some vertical lines joining the limits of the slider intervals define a polygonal area (called Satisfying Area) corresponding to the elements E_i satisfying the query (yellow area in Figure 1). Concentric areas are drawn around the central one (green areas in Figure 1). They are defined as the union of two polygons (left and right hand side of the Satisfying Area) also spanning all criteria. They are associated to decreasing degrees of satisfaction of the query. This approach relies on and extends Otjacques et al's work [OCSF11] to queries composed of $n > 2$ criteria. Nevertheless, in our approach these polygonal areas only help to create a visual metaphor. The colored polygons should be considered as a legend integrated within the query component and not as a display

space for the data. Therefore, contrary to the planar layout, a specific point inside these polygons cannot be directly mapped to an element E_i.

Concentric Areas Computation

In this more formal section, CA_k refers to the k^{th} Concentric Area (CA_1 is the closest Concentric Area around the Satisfying Area). $CIL_{k,j}$ refers to CA_k's Left Interval of values for the criterion j and $CIR_{k,j}$ to the Right one (cf. Figure 1). The left $d_{left,j}$ and right $d_{right,j}$ distances are set by the user for each criterion j, which allows to compute all intervals $CIL_{k,j}$ and $CIR_{k,j}$ and subsequently to draw the areas CA_k.

$$CIL_{k,j} = [a_j - k.d_{left,j}\ ,\ a_j - (k-1).d_{left,j}]$$

$$CIR_{k,j} = [b_j + (k-1).d_{right,j}\ ,\ b_j + k.d_{right,j}]$$

The user can choose between two computation modes for $d_{left,j}$ and $d_{right,j}$. They are either defined (1) as a fixed value (e.g. *100 EUR*) or (2) as a percentage of the closest interval limit (e.g. *10%* of the lower limit of the related criterion). The left and right distances can be computed with different modes. For instance, for a criterion about the cost of the data elements, $d_{left,cost}$ can be computed as *10%* of the left limit and $d_{right,cost}$ can be set to *30 EUR* (independent from the right limit value). The elements E_i that fully satisfy all the query criteria are associated to the Satisfying Area. Each remaining element is associated to the Concentric Area CA_k where k is obtained with the following algorithm:

for k: max..1
  for each criterion j
    if $\exists\ x^*_{i,j} \in CIL_{k,j} \cup CIR_{k,j}$ *then return k*

Figure 2: *The query zone pilots the data view (TreeMap).*

In other words, we identify for each element E_i what is the least satisfied criterion, i.e. the attribute whom the value of E_i is the most distant from the interval of acceptable values. This criterion is chosen to identify the Concentric Area that E_i will belong to. This Concentric Area is easily found by computing in which interval $CIL_{k,j}$ or $CIR_{k,j}$ is located the attribute value of E_i. Placing the elements into a given CA_k is similar to computing a DOI function. However, our component makes visually explicit how this DOI function is computed (which is often hidden in other systems). For instance, asymmetry in Concentric Areas specification (meaning asymmetric user sensitivity to some attributes) can be instantly perceived.

The Concentric Areas are painted with a monochromatic scale (green areas in Figure 1). The elements E_i displayed in the data view are painted in the same color as the Concentric Area color they belong to (cf. Figure 2).

Note that this metric is very conservative because it does not consider the number of satisfied criteria to link an element to a Concentric Area but is based on a "worst case" approach.

4. Implementation

First of all, the Multidimensional Concentric Sliders (MCS) component is designed to pilot a data view. The information about the dataset that it may provide (see next section) should only be considered as a support to the querying specification. The basic rule of the user interface is that the MCS component defines a mapping between some degrees of satisfying the criteria of a query and a monochromatic scale. Since the related colors are used in the data view(s), we can only link the query component to data views where the elements E_i can be colored. (e.g. TreeMap in Figure 2).

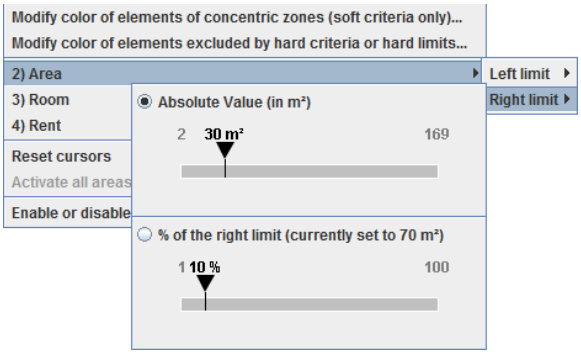

Figure 3: *Contextual menu.*

The user interface adopts the direct manipulation paradigm. When a cursor in the query component is moved the data view is automatically updated with the new appropriated colors (update time < 1 second). By default the query component includes the list of all attributes of the elements E_i. The user can of course limit this list to a subset of the attributes.

Via a contextual menu (Figure 3) the user can choose which mode is used to compute $d_{left,j}$ and $d_{right,j}$ for each

criterion. A modification of the computation method is instantly propagated to the query component and to the data view(s) to benefit from the direct manipulation advantages. For instance, when the user moves the slider setting the right limit of the *Area* attribute, he sees the related Concentric Area changing in real time. Of course, the user can also modify the colors respectively associated to the Satisfying Area, the Concentric Areas and the excluded elements.

When the number of criteria increases, the query component included in the main window of the application (cf. Figure 2) may become difficult to use. To overcome this limitation, the user can simply double-click on the title of the query component to display it in a larger separate window where some extra features are enabled (cf. Figure 4).

The basic features of the MCS component discussed above have been completed by other ones in order to offer a better support to the user (cf. Figure 4).

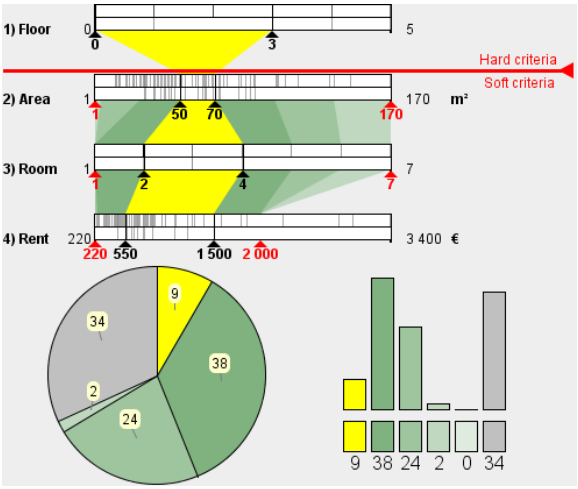

Figure 4: *MCS implementation.*

A first complementary idea is that the user accepts to relax his constraints for some criteria (which motivates the use of Concentric Areas) but not for others. We have then introduced the concepts of soft and hard criteria. Soft criteria are the ones that the user can accept to adapt (e.g. because he cannot express a strict limit or because the criteria is not very important for him). As a consequence, we need to give him some information about how far from the Satisfying Area the elements are. Hard criteria are criteria that the user wants to be strictly respected. In this case the concept of Concentric Area is irrelevant. The elements E_i that do not satisfy a hard criterion are neither in the Satisfying Area nor in any Concentric Area. A red line is drawn on the query component: criteria above this line are considered as hard criteria and the ones below are considered as soft criteria (e.g. in Figure 4 *Floor* is a hard criterion and *Area* is a soft criterion). In order to specify the nature of the criteria (hard vs. soft) the user can either vertically move the red line or move the criteria. Note that the order of the criteria above or below the red line is not meaningful.

A second improvement consists in extending the notion of hard/soft constraints to the limits of the criteria. Two red cursors are then added to the slider to visualize some absolute limits. Values respectively lower or higher of these red points are formally excluded by the user. (e.g. in Figure 4 the rent cannot exceed *2000 EUR*). Consequently the Concentric Areas are drawn from the Satisfying Area to the absolute limits but never go beyond them (e.g. in Figure 4 there is no Concentric Area for the rent values higher than *2000 EUR*). If an element E_i has an attribute value exceeding an absolute limit it is placed in the worst area regarding the satisfaction of the query (colored in grey in this configuration).

Third, we have also followed Spence and Tweedie's recommendation to visualize sensitivity information [ST98]. Each slider is composed of two horizontal bars. In the top bar, some vertical lines show the distribution of the elements for the related attribute (complete dataset considered). In the bottom bar, the vertical lines visualize the distribution of the elements that satisfy the other criteria (only elements completely satisfying the query Q). This feature helps to know before moving a cursor if the Satisfying Area will gain or lose some elements (e.g. in Figure 4 at least one apartment can be added to the Satisfying Area by moving the right *Floor* cursor to *5 floors* but there will be no effect by moving the right *Room* cursor to *6 rooms*).

The next feature is mainly useful for supporting interaction. If the number of Concentric Areas is high, the user may face some difficulties to distinguish the elements belonging to one of them (e.g. show only the elements of the first Concentric Area CA_1, i.e. the elements that just fail to satisfy all the criteria). The coloration of any area can be disabled/enabled in the query component and in the data view. All disabled areas are colored in grey, which helps to identify the elements that belong to the active ones.

We have also added a pie chart and an histogram to show how the elements E_i are distributed among the various areas (Satisfying Area, Concentric Areas, disabled areas). The number of elements in each area is displayed in both graphics which use the same coloring strategy as the core query component. For instance, in Figure 4, 9 elements satisfy all query criteria and 38 elements are second choice candidates.

5. Conclusions

This paper proposes a new component to visually specify queries combining criteria that are differently considered by the user. He can express how strictly the threshold values should be respected. Further work will investigate some pending problems (e.g. disjunctive queries, categorical attributes). We also plan to collect feedback from pilot users regarding the MCS component.

References

[AWS92] AHLBERG C., WILLIAMSON C., SHNEIDERMAN B.: Dynamic Queries for Information Exploration: an Implementation and Evaluation. In *CHI '92 Proceedings of the SIGCHI conference on Human factors in computing systems* (May 1992), pp. 619–626. doi:10.1145/142750.143054. 2

[DH02] DOLEISCH H., HAUSER H.: Smooth Brushing For Focus+Context Visualization Of simulation Data In 3D. In *Journal of WSCG* (2002), pp. 147–154. 2

[HLD02] HAUSER H., LEDERMANN F., DOLEISCH H.: Angular Brushing of Extended Parallel Coordinates. In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis 2002)* (2002), pp. 127–130. doi:10.1109/INFVIS.2002.1173157. 2

[HS04] HOCHHEISER H., SHNEIDERMAN B.: Dynamic Query Tools for Time Series Data Sets, Timebox Widgets for Interactive Exploration. *Information Visualization 3*, 1 (Nov 2004), 1–18. doi:10.1145/993176.993177. 2

[Huo08] HUO J.: KMVQL: a Visual Query Interface Based on Karnaugh Map. In *AVI '08 Proceedings of the working conference on Advanced visual interfaces* (May 2008), pp. 243–250. doi:10.1145/1385569.1385609. 1

[HY06] HOEBER O., YANG X. D.: The Visual Exploration of Web Search Results Using HotMap. In *10th International Conference on Information Visualisation (IV'06)* (July 2006), pp. 157–165. doi:10.1109/IV.2006.108. 1

[Jon98] JONES S.: Graphical Query Specification and Dynamic Result Previews for a Digital Library. In *UIST '98 Proceedings of the 11th annual ACM symposium on User interface software and technology* (Nov. 1998), pp. 143–151. doi:10.1145/288392.288595. 1

[KK94] KEIM D., KRIEGEL H.-P.: VisDB: Database Exploration Using Multidimensional Visualization. *IEEE Computer Graphics and Applications 14*, 5 (Sept./Oct. 1994), 40–49. doi:10.1109/38.310723. 1, 2

[OCSF11] OTJACQUES B., CORNIL M., STEFAS M., FELZ F.: Concentric Sliders to Display Partial Satisfaction of Query Criteria. In *15th International Conference on Information Visualisation (IV'11)* (July 2011), pp. 59–64. doi:10.1109/IV.2011.9. 2

[Spo93] SPOERRI A.: InfoCrystal: a visual tool for information retrieval & management. In *CIKM '93 Proceedings of the second international conference on Information and knowledge management* (Nov. 1993), pp. 11–20. doi:10.1145/170088.170095. 1

[ST98] SPENCE R., TWEEDIE L.: The Attribute Explorer: information synthesis via exploration. *Interacting with Computers 11*, 2 (Dec. 1998), 137–146. doi:10.1016/S0953-5438(98)00022-8. 2, 4

[WS92] WILLIAMSON C., SHNEIDERMAN B.: The Dynamic Homefinder: evaluating dynamic queries in a real estate information exploration system. In *SIGIR '92 Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval* (June 1992), pp. 338–346. doi:10.1145/133160.133216. 2

[YS93] YOUNG D., SHNEIDERMAN B.: A graphical filter/flow representation of Boolean queries: A prototype implementation and evaluation. In *Journal of the American Society for Information Science* (July 1993), pp. 327–339. doi:10.1002/(SICI)1097-4571(199307)44:6<327::AID-ASI3>3.0.CO;2-J. 1