VOYAGER: Automatic Computation of Visual Complexity and Aesthetics of Graph Query Interfaces

Duy Pham Nanyang Technological University Singapore duy4@ntu.edu.sg

ABSTRACT

People prefer attractive visual query interfaces (VQI). Such interfaces are paramount for enhancing usability of graph querying frameworks. However, scant attention has been paid to the *visual complexity* and *aesthetics* of graph query interfaces. In this demonstration, we present a novel system called VOYAGER that leverages on research in computer vision, human-computer interaction (HCI) and cognitive psychology to automatically compute the visual complexity and aesthetics of a graph query interface. VOYAGER can not only guide VQI designers to iteratively improve their design to balance usability and aesthetics of visual query interfaces but it can also facilitate quantitative comparison of the visual complexity and aesthetics of a set of visual query interfaces. We demonstrate various innovative features of VOYAGER and its promising results.

1 INTRODUCTION

A recent survey [13] revealed that graph visualization, graph query languages, and usability are considered as some of the top challenges for graph processing. Although considerable efforts have been invested toward efficient and scalable processing of graphs, the above issues have received lesser attention from the data management community. In particular, a starting point for addressing the usability and visualization challenges is the deployment of a visual query interface (vQI) that can enable an end user to (a) *draw* a graph query interactively in lieu of writing it using a graph query language and (b) visualize and explore the result matches effectively in a user-friendly manner. Figure 1 is an example of such a vQI (encapsulated by the red rectangle).

The visual appearance of a visual interface (*i.e.*, aesthetics) impacts its usability as it influences the way users interact with it (*i.e.*, *aesthetic-usability effect*¹). Research in HCI and psychology reveal that the *visual complexity* of an interface plays a pivotal role in this context as it decreases aesthetics and usability [6, 14] and increases cognitive load [4]. Several studies have found a strong relationship between aesthetic preferences and visual complexity [2, 5, 11]. According to Berlyne's aesthetic theory [2], the relationship between them follows an inverted U-shaped curve where stimuli of a moderate degree of visual complexity are considered pleasant but both less and more complex stimuli are considered unpalatable.

Visual complexity of an interface is operationally defined as the combination of different features such as *quantity of information, variety of visual form, spatial organization,* and *perceivability of details* [8]. *Quantity of information* is the most common facet of visual complexity: the more units of information are on the screen, Sourav S Bhowmick Nanyang Technological University Singapore assourav@ntu.edu.sg

the more complex it appears to the users. *Variety of visual form* embodies the number of colors, shapes, sizes, background textures and other visual features used to represent information. It is shown to increase complexity when their numbers increase [12]. *Spatial organization* refers to "the tendency of human perception to see structural repetition and regular positioning as simplifying presented information" [8]. Intuitively, poor spatial organization impacts visual complexity adversely. Lastly, *perceivability of details* [6] reflects the limitations of human visual perception, such as needing to use the focal vision to perceive fine-grain detail (*i.e.*, visual congestion) or grappling to efficiently distinguish lowcontrast items from background. Importantly, a single feature (*e.g.*, quantity of information) is insufficient in understanding visual complexity [8]. Otherwise, an empty visual interface will be considered as a best design w.r.t. visual complexity.

Existing research on visual complexity and aesthetics primarily focus on generic visual interfaces and web sites [5-8, 11]. Visual query interfaces for graphs typically have distinct content and structure from them (detailed in Section 2). In this demonstration, we present a novel visual complexity and aesthetics² evaluation tool for graph query interfaces called VOYAGER (Visual COmplexitY and Aesthetics of Graph QuEry InteRface). The key benefits of such a tool are at least two-fold. First, it allows a visual graph query interface designer to balance between the aesthetic appearance of a voi and its visual complexity by incrementally refining the interface design based on feedback received from the complexity and aesthetic scores. This will minimize the effort to create aesthetically pleasing vois by ensuring that they can still be aesthetically attractive but also usable and not overloaded with information that increases visual complexity. Consequently, it facilitates the creation of usable and attractive vois for graph querying. Note that VOYAGER is particularly useful for small- and medium-sized companies and individual developers as they often may not have the budget to hire design agencies or experts in these fields. Hence, they may have to do their voi design themselves. Second, it can be used to compare and evaluate the visual complexity or aesthetics of a set of vois in a more systematic manner that can complement subjective evaluation by humans.

Given an image (*i.e.*, screenshot) of a vQI for graphs, VOYAGER analyses the structure and content of different *panels* of the interface using image processing techniques to compute its *complexity score*, which is inspired by the visual complexity computation of web sites [5]. Note that we analyze the vQI screenshot instead of its underlying source code as the former is a better representation of what a user sees. The *quantitative model* utilized by VOYAGER to compute visual complexity is inspired by regression-based models used for computing visual complexity of web sites [5] (for reasons justified later). Furthermore, it quantifies the *aesthetic score* of an interface by utilizing the corresponding visual

815



 $^{^{1}} https://www.nngroup.com/articles/aesthetic-usability-effect/\\$

^{© 2023} Copyright held by the owner/author(s). Published in Proceedings of the 26th International Conference on Extending Database Technology (EDBT), 28th March-31st March, 2023, ISBN 978-3-89318-092-9 on OpenProceedings.org. Distribution of this paper is permitted under the terms of the Creative Commons license CC-by-nc-nd 4.0.

 $^{^2\}rm We$ acknowledge that aesthetics is a complex phenomenon consisting of many culture-independent and culture-specific facets. Our tool aims to only address culture-independent facets, related to perceived visual complexity. This effort is closely relate to aesthetic pleasure across all cultures [10].



Figure 1: vQI (red rectangle) of Pubchem in VOYAGER.

complexity score and Berlyne's aesthetic theory [2]. In summary, VOYAGER is a multi-disciplinary framework bridging graph query interface design with image processing techniques from computer vision and visual complexity and aesthetics computation models from the domains of HCI and cognitive psychology.

In this demo, we shall first present a walk-through of the VOYAGER tool and show how it can quantify the visual complexity and aesthetic scores of a vQI simply by clicking and dragging an image of the visual interface. Then, we will show how it can provide explanations to the contributions of different components of the vQI to its visual complexity. Finally, we will demonstrate how VOYAGER can be used to systematically compare the visual complexity and aesthetics of a set of vQI and guide in designing usable and aesthetically pleasing interfaces.

2 SYSTEM OVERVIEW

In this section, we describe the system architecture of VOYAGER. We begin by presenting the generic structure of a visual graph query interface, which we shall be using subsequently.

2.1 Structure of Graph Query Interfaces

Our initial investigation of several real-world visual interfaces for graph query construction reveal that these interfaces typically share the following key panels: (1) A *Menu Panel* to display a list of items and buttons related to graph query formulation and processing. (2) A *Label Panel* to display a set of labels or attributes of nodes or edges of the underlying data. (3) A *Pattern Panel* to display a set of canned patterns (*i.e.*, small connected subgraphs) that can aid query formulation. (4) A *Query Panel* for constructing a graph query graphically by adding a node or canned pattern iteratively. (5) A *Results Panel* that displays the query results.

Note that the *Label* and *Pattern* Panels are optional as some interfaces may not include them. Figures 1 and 2 depict screenshots of *Pubchem* (http://pubchem.ncbi.nlm.nih.gov/) and *eMolecule* (https://www.emolecules.com/) vQIs, respectively, for querying a set of chemical compounds. Observe that the menu items, canned patterns (shown using a blue border in the vQI), and labels of nodes (shown using a green rectangle) are depicted using images and text. Also, the *Query Panel* is an empty space for all vQIs and the *Result Panel* is often visible only after a query is executed. Lastly, these vQIs typically present a single user interface for formulating queries. Hence, we shall assume a single screenshot of the vQI for a given database. Nevertheless, voyAGER can be easily extended to handle multiple vQIs by computing the visual complexities of corresponding screenshots iteratively.

2.2 System Architecture

Figure 3 depicts the architecture of VOYAGER and mainly consists of the following modules.



Figure 2: vQI of eMolecules (in red rectangle).



Figure 3: Architecture of VOYAGER.



Figure 4: GUI of VOYAGER.

The GUI module. Figure 4 is a screenshot of the visual interface of VOYAGER. It consists of two panels. Panel 1 enables us to load a screenshot of a visual graph query interface in the form of an image file (png or jpeg format). Once a screenshot is loaded, it is displayed in this panel. Panel 2 is used to compute the visual complexity and aesthetics of the screenshot by clicking on the Process Image button. The results (complexity and aesthetic scores) are then displayed in this panel along with *structural information* of the vQI. Figures 1 and 2 show two examples of the contents of Panels 1 and 2 upon clicking on the Process Image button. Clicking on the Explanation button invokes the *Explanation* module that enables a user to understand the contributions of different components of the vQI to the complexity score.

The Denoising module. The goal of this module is to clean the image of the vQI by (partially) removing noise for further processing. Image noise is random variation of brightness or color information in images. It leverages on the *Gaussian Blur* technique³ in computer vision to remove the noise and to enhance image structure for subsequent image segmentation. Observe that the *Denoising* module primarily targets the *Menu, Label,* and *Pattern Panels* of the vQI since the *Query Panel* typically does not display much variation in brightness or color as it is empty.

 $^{^3 \}rm https://docs.opencv.org/2.4/modules/imgproc/doc/filtering.html?highlight=gaussianblur# gaussianblur$



Figure 5: TLCs in eMolecules vQ1.

The VQI Segmentation module. Given the denoised image of the vQI, the goal of this module is to segment it in order to facilitate the identification of objects and boundaries for the subsequent computation of visual complexity. To this end, it leverages on an image segmentation technique. The result of image segmentation is a set of segments that collectively cover the entire image or a set of contours extracted from the image. In this demonstration, we use the *Canny edge detection* method⁴, a widely popular technique for reliable detection of boundaries of objects within an image. Observe that this module also focuses on the *Menu, Label* and *Pattern Panels* as in most vQIs the visual objects for query formulation are localized in these panels.

The Visual Object Detection module. Michailidou *et al.* [5] have reported that the visual complexity of a web page is influenced by the number of images, text and *top left corner* (TLC) but not by menu items. Intuitively, a TLC is the top left corner of a *block*. A *block* is a box that is surrounded by white space only. Intuitively, a web page consists of a set of sections and can be divided into boxes (a box is an area enclosed by four lines). If the left and top sides of a box are not adjacent to or share a common side with another box, then it is counted as a TLC.

In contrast to web pages, a voi for graphs typically contains a Menu, Label, and Pattern panels. Consequently, the content of these structural elements and their characteristics (e.g., size, color) can be used to determine the visual complexity of a voi. Since visual objects in these panels are typically images and text elements, we advocate that these elements can be exploited to this end. Specifically, images in the form of graphical icons typically appear in a VQI as they are used to construct queries. Similarly, text data may appear in these panels. This includes text in buttons and within images. Lastly, since a voi can be divided into multiple panels and each panel may contain a set of visual objects, we can consider a block to be a group of images or text in close proximity. The visual distinction of each block are made with the use of colors, tables, lines or spacing. Then a TLC in a VQI can be considered as a block's top left corner. Figure 5 shows the five TLCs in the *eMolecules* VOI (encapsulated by green borders).

The goal of this module is to identify the text and image objects as well as TLCs from the preprocessed image of the vol. It first identifies all the visual objects and then checks for each object whether it is a text or an image. To detect visual objects, it utilizes the findContours() procedure of *OpenCV* library (https://opencv. org/). Next, for each detected object, this module leverages on an optical character recognition (OCR) tool (we use the pythontesseract) to identify the text embedded in images. In order to identify the TLCs, this module implements a VQI chunk rendering algorithm which groups adjacent elements (graphical icon, button, text) into a block. These blocks are dilated by filling them in white color. Subsequently, the TLCs are identified from these blocks.

The Visual Complexity Computation module. This module is responsible for computing the visual complexity score of a vQI. As visual complexity is a continuous variable, a regression rather than a classification is a better choice to model it. Michailidou *et al.* [5] quantified the visual complexity of a web page using regression-based analysis and showed that only the number of TLCs, words, and images are significant. Since vQIs also contain words, images, and TLCs, we can leverage on this regression model to design the visual complexity computation model for vQIs. However, we cannot directly adopt it as the structural characteristics of vQIs are different from a web page.

A web page typically consists of a large number of words and images can be organized in a wide variety of ways. The number of TLCS can vary widely. In contrast, in a visual graph query interface, images are typically clustered together in the *Label*, *Pattern*, and *Menu Panels*. The words are typically visible in the *Label* or *Menu Panel* and often they may be overlaid on images. The number of TLCS is also limited in a VQI (*e.g.*, only five in Figure 5). Hence, the *complexity score* is computed as follows:

$$V_G = C + m_1 \sum_{j=1}^{n_t} i_d(j) + m_2 \sum_{j=1}^{n_t} w_d(j) + m_3 \times n_t$$

In this equation, n_t is the number of TLCS, i_d and w_d denote image density and word density, respectively. The image density is the ratio of the number of images in a TLC j and its area (approximated by a rectangle). That is, $i_d(j) = \frac{n_i^j}{length_j \times width_j}$. Similarly, the *word density* is the ratio of the number of words in a TLC and its area. That is, $w_d(j) = \frac{n_w^j}{length_j \times width_j}$. Observe that the larger is the image (resp. word) density, the more number of images (resp. words) are packed in a TLC. Furthermore, the font size and image size decrease and the contour congestion increases with increasing density. Consequently, a larger density increases the visual clutter of the interface, leading to a higher visual complexity⁵. Note that m_1 , m_2 and m_3 can be set based on relative importance of these three components. In our demonstration, we allow users to vary all these constants and explore their impact on the complexity score. Note that we ignore the color variability measures [7] as our investigation with several real-world vois reveal that typically very few dominant colors are used in their design (e.g., two in Figure 5).

We emphasize that Michailidou *et al.* [5] use only the number of TLCs, words and images in a page to compute the visual complexity. In contrast, we use image and word densities to accurately capture the structural characteristics of VQIS.

Observe that in our visual complexity computation we ignore the topology of a canned pattern in an image. This is primarily because these patterns are small-sized planar graphs (*e.g.*, triangle, chain, rectangle, ring) in most real-world interfaces (*e.g.*, Figures 1, 2). Hence, their cognitive cost is negligible [15]. We also ignore the menu items in a vQI that may be invoked when a user clicks on certain icons. This is because menu items do not significantly influence the visual complexity of interfaces [5–7].

⁴https://docs.opencv.org/3.1.0/da/d22/tutorial_py_canny.html

⁵Eye movement research reports that the human eye is capable of focusing on only a small area at one time - which is referred to as *perceptual span*. Hence, a higher density leads to more objects for a human eye to focus and the cognitive load of identifying and separating these objects.

Lastly, we ignore semantics of the images (*e.g.*, the periodic table in *Pubchem*) in computing the complexity score. Although some domain experts may be familiar with the semantics of the images, this is not the case for non-expert users (a query writer is not necessarily a chemist). More importantly, familiarity with semantics only improves the task complexity but not necessarily the visual complexity, which is an orthogonal concept.

The Aesthetics Computation module. This module is responsible for computing the *aesthetic score* of the vQI by utilizing the complexity score. Specifically, it leverages on Berlyne's aesthetic theory [2] to relate the complexity and aesthetic scores using an inverted-U function. We depict the aesthetic score using stars in Panel 2 of the voyAGER interface. Figures 1 and 2 show the scores of *Pubchem* and *eMolecules*, respectively.

The Explanation module. For a deeper insight to the visual complexity of a vQI, this module highlights components of the vQI (TLCS, images and words) that contribute to the complexity score. It also explains the reasons behind the complexity score. This facilitates a developer to comprehend the components of a vQI that impact the visual complexity and improve the design accordingly.

2.3 Performance Summary

To understand the performance of VOYAGER, we first compute and rank the complexity and aesthetic scores of the *Pubchem*, *eMolecules*, and *Drugbank* (go.drugbank.com/structures/search/ small_molecule_drugs/structure) vQIs using it. Next, we ask a set of volunteers to provide their subjective feedback on these interfaces w.r.t. their visual complexity and aesthetics and rank them accordingly. Lastly, we compare these two rankings.

Table 1 reports the output of VOYAGER. The image and word densities are largely impacted by the numbers of images, words and TLCs since the areas are similar in the three interfaces. *Pub-chem* (Figure 1) has the highest complexity score and the lowest aesthetic score. It has a large number of words (76) and images (27) but there is only one TLC. On the other hand, *eMolecules* (Figure 2) reports the lowest (resp. highest) complexity (resp. aesthetic) score. It has a significantly lower number of words (7) and images (28) and the number of TLCs is 5.

We invited 16 unpaid volunteers (undergraduate and graduate students). We ask them to rank these three interfaces based on their increasing *perceived* visual complexity and aesthetics. Our results show that all volunteers rank *Pubchem* to be the most visually complex (*i.e.*, rank 3) and aesthetically least pleasing (*i.e.*, rank 1). 14 volunteers ranked *eMolecules* and *Drugbank* 1 (resp. 3) and 2 (resp. 2), respectively, for visual complexity (resp. aesthetics). Two volunteers ranked *Drugbank* first for visual complexity as they ignored the different search options in the vQI. Overall, *the rankings of the volunteers are highly consistent with the output of VOYAGER*.

3 RELATED SYSTEMS AND NOVELTY

To the best of our knowledge, automatic computation of visual complexity and aesthetics of graph query interfaces have not been studied in the data management community [3]. Most germane to our work are efforts in the HCI community [5–7, 11]. These efforts focus on web sites/pages. In particular, in addition to visual clutter, contour congestion and layout quality, [6, 7] consider additional metrics such as color variability and prototypicality (*i.e.*, amount to which an object is representative of a class of objects). vQIs are different from web pages/web sites

Table 1: Performance of VOYAGER

Visual query interface	Complexity score	Aesthetic score
Pubchem	5.87	2
Drugbank	4.58	3
eMolecules	3.82	5

w.r.t. structure and content. Hence, as remarked earlier, these techniques cannot be directly adopted for graph query interfaces effectively as they do not utilize the unique characteristics of vQIS. Furthermore, web sites have more variability in layout and content compared to vQIS. Consequently, certain metrics such as protypicality are not relevant to vQIS.

4 DEMONSTRATION OBJECTIVES

VOYAGER is implemented in Python. Our demonstration will be loaded with a few real vQIs from academia and industry (*e.g., Pubchem, eMolecules, Drugbank*). The audience can also provide screenshots of their own vQIs.

One of the key objectives of the demonstration is to enable the audience to interactively experience the computation of visual complexity and aesthetics of a vQI in real-time (all interfaces can be processed within 10s in vOYAGER). Specifically, the GUI of vOYAGER shall assist users in gaining such experience. Through the *Visual Complexity* and *Aesthetics Computation* modules, one will be able to view the complexity and aesthetic scores of the selected vQI. The audience can interactively comprehend the contributions of various components of the vQI to the visual complexity in real-time by clicking on the Explanation button (Figure 1). Furthermore, one will be able to *compare* the aesthetics of a set of selected vQIs and explore if the results generated by VOYAGER are consistent with their impression of the interfaces.

REFERENCES

- A. De Angeli, et al. Interaction, Usability and Aesthetics: What Influences Users' Preferences? In Conf. on Designing Interactive Systems, 2006.
- [2] D. Berlyne. Studies in the new Experimental Aesthetics. Washington D.C., Hemisphere Pub. Corp., 1974.
- [3] S. S. Bhowmick, B. Choi. Data-driven Visual Query Interfaces for Graphs: Past, Present, and (Near) Future. In SIGMOD, 2022.
- [4] S. Harper, C. Jay, E. Michailidou, H. Quan. Analysing the Visual Complexity of Web Pages using Document Structure. *Behaviour & Information Technology*, 32(5), 2013.
- [5] E. Michailidou, S. Harper, S. Bechhofer. Visual Complexity and Aesthetic Perception of Web Pages. *In SIGDOC*, 2008.
- [6] A. Miniukovich, A. De Angeli. Quantification of Interface Visual Complexity. In AVI, 2014.
- [7] A. Miniukovich, A. De Angeli. Computation of Interface Aesthetics. *In SIGCHI*, 2015.
- [8] A. Miniukovich, S. Sulpizio, A. De Angeli. Visual complexity of graphical user interfaces. In AVI, 2018.
- [9] A. Oliva, M. L. Mack, M. Shrestha, A. Peeper. Identifying the Perceptual Dimensions of Visual Complexity of Scenes. In Proc. of the 26th Annual Meeting of the Cognitive Sc. Society, 2004.
- [10] R. Reber. Processing Fluency, Aesthetic Pleasure, and Culturally Shared Taste. In Aesthetic Science: Connecting Minds, Brains, and Experience, 2012.
- [11] K. Reinecke, et al. Predicting Users' First Impressions of Website Aesthetics with a Quantification of Perceived Visual Complexity and Colorfulness. In SIGCHI, 2013.
- [12] R. Rosenholtz, Y. Li, L. Nakano. Measuring Visual Clutter. Journal of vision, 7(2), 2007.
- [13] S. Sahu, A. Mhedhbi, S. Salihoglu, J. Lin, M. T. Özsu. The Ubiquity of Large Graphs and Surprising Challenges of Graph Processing. *PVLDB*, 11(4), 2017.
- [14] A. N. Tuch, E. E. Presslaber, M. Stöcklina, K. Opwis, J. A. Bargas-Avila. The Role of Visual Complexity and Prototypicality Regarding First Impression of Websites: Working Towards Understanding Aesthetic Judgements. *International J.* of Human-Computer Studies, 70, 2012.
- [15] C. Ware, H. Purchase, L. Colpoys, M. McGill. Cognitive Measurements of Graph Aesthetics. *Information Visualization*, 1(2), 2002.