Special Interest Group on Visualization Grammars

Xiaoying Pu
xpu@umich.edu
University of Michigan
Ann Arbor, Michigan, USA

Matthew Kay
mjskay@northwestern.edu
Northwestern University
USA

Steven Drucker*
sdrucker@microsoft.com
Microsoft Research
USA

Jeffrey Heer*
University of Washington
USA

Dominik Moritz*
Carnegie Mellon University
USA
Apple Inc.
USA

Arvind Satyanarayan*
Massachusetts Institute of Technology
USA

ABSTRACT

Visualization grammars, often based on the Grammar of Graphics, are popular choices for specifying expressive visualizations and supporting visualization systems. However, there are still open questions about grammar design and evaluation not well-answered in visualization research. In this SIG, we propose to discuss what makes a grammar "good" and explore evaluation methodologies best suited for visualization grammars.

CCS CONCEPTS

• **Human-centered computing** → **Visualization design and evaluation methods**; **Visualization theory, concepts and paradigms**; Visualization application domains.

KEYWORDS

Grammar of Graphics

ACM Reference Format:
Xiaoying Pu, Matthew Kay, Steven Drucker, Jeffrey Heer, Dominik Moritz, and Arvind Satyanarayan. 2021. Special Interest Group on Visualization Grammars. In *CHI Conference on Human Factors in Computing Systems Extended Abstracts (CHI '21 Extended Abstracts), May 8–13, 2021, Yokohama, Japan.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3411763.3450406

1 INTRODUCTION

The visualization research community has come up with many visualization grammars or domain-specific languages based on the *Grammar of Graphics* (GoG) [20]. These grammars let users specify a wide range of visualizations. Examples include D3 [2] that generates expressive, interactive web-based charts; Vega-Lite [14] and `ggplot2` [18] make exploring alternative designs of statistical graphs easy. In the commercial space, Tableau is a successful grammar-based system that stemmed from Polaris [16].

The popularity of these visualization grammars is evident. As open-source packages, `ggplot2` (R) has around 1.5 million downloads from CRAN per month[1] and Vega-Lite (JSON specification) has 2 million CDN hits per month[2]. Users of visualization grammars are well beyond the visualization research community — journalists use D3[3] and `ggplot2` [5] in their reporting, and authors of `ggplot2` books have background in fields like statistics [19] and sociology [4].

Increasingly, visualization grammars have become building blocks for other formalisms and systems. To support data exploration and question answering in visual analysis, Voyager 2, a mixed-initiative system, is powered through CompassQL, "a generalization of the Vega-Lite grammar" [14, 22]. To depict probability distributions, *A Probabilistic Grammar of Graphics* makes probability expressions such as $P(A|B)$ first-class citizens on top of the original GoG [13]. For high-performance visual analytics, P6 leverages a declarative grammar to integrate interactive visualizations with machine learning algorithms [8]. Gemini [6] and Canis [3] are two grammars that create animated transitions in statistical graphics and charts. These emerging applications in exploring and communicating data depend on well-designed grammar abstractions.

Given the popularity and significance of visualization grammars, there are still fundamental questions about grammar design and evaluation unanswered in visualization research. Therefore through this SIG, we want to take stock from previous works and reflect on what makes a grammar "good", including thoughts about the design choices, evaluation methods, and usability of the grammars.

2 BACKGROUND: GRAMMAR OF GRAPHICS

The power of visualization grammars lies in their abstractions based on the Grammar of Graphics (GoG) [20]. The original GoG defines a set of components that make up a visualization, including data, aesthetics (encoding channels such as x, y coordinates and size), statistics (data transformation), geometry (mark types such as points and bars), scale, coordinate, and facet. The GoG is flexible; for example, we can make a small change in the specification of a scatter plot to produce a bar chart (geometry(*point*) → geometry(*bar*)), keeping everything else the same. Though the original GoG has

*Listed alphabetically by last name.

CHI '21 Extended Abstracts, May 8–13, 2021, Yokohama, Japan

ACM ISBN 978-1-4503-8095-9/21/05.
https://doi.org/10.1145/3411763.3450406

[1] As of December, 2020 per https://cranlogs.r-pkg.org/
[2] As of December, 2020 per https://www.jsdelivr.com/package/npm/vega-lite
[3] Collection of D3 visualizations by the New York Times: https://github.com/d3/d3/wiki/Gallery#the-new-york-times-visualizations

no software implementation, GoG-style visualization specifications are readily expressed in JSON (e.g. Vega [4]) and in object-oriented programming (e.g., `ggplot2` [18]). With the growing need to visualize data on the web and in computational notebooks, visualization grammars are appealing choices, for they are abstract enough to make the specifications concise and easy to modify, but also flexible enough to produce many custom visualization designs.

3 PROPOSED TOPICS TO DISCUSS

3.1 What makes a grammar "good"

How can grammar designers know that a grammar is "good"? Current works evaluate grammars based on expressiveness (expressing a wide range of visualizations [9]) and generativeness (generating new visualizations) and in terms of the cognitive dimensions of notations [1], in addition to usage and adoption ([10, 21]). But we should also consider what it means to implement a grammar well and how best to create abstractions that work within different contexts.

3.1.1 What makes a grammar? In visualization literature, we have been using the term *grammar* without explicitly articulating what we mean by it. Recent papers have described grammars for interactive visualizations [14], probability distributions [13], and animated transitions [3, 6], in addition to GoG-inspired systems in industry (*e.g.* Plotly Express [12]). These new grammars have gone beyond the scope of the original GoG, and we should benefit from a precise definition of visualization grammar, explicating what purpose should a grammar serve, what standards should it meet, and what benefits we can expect from designing a grammar.

3.1.2 Handling grammar rules and edge cases. When creating a new grammar, designers need to set grammar rules. Those rules might need to cover many edge cases that come from the combinatorics of grammar components. For example in `ggplo2`, an R library based on GoG, the documentation[5] lists 30 default geometries, six position adjustments, four types of aesthetics, 21 types of scales, and six coordinate systems. When a user combines those grammar components to build a `ggplot2` visualization, they are afforded with thousands of possible combinations, though not all are valid. Grammar specification can even be recursive, as in the case of ATOM, a grammar for creating unit visualizations inspired by GoG and L-systems [11]. How does the grammar designer or developer know with certainty that all combinations can produce valid visualizations, or if a combination is incorrect, error handling is in place? What do bugs look like in a visualization specification? A grammar also poses interesting constraints on continuing development – if one grammar component is faulty, its error surface is combinatorial relative to standalone components.

3.1.3 Visualization grammar for different use cases. Visualization grammars can be used for explaining/exploring data and supporting visualization systems. With these use cases, we might want to synthesize what makes for effective abstractions or grammar components and differentiate them from mere configurations or settings. For example, one criterion for a good grammar can be, "Can the specifications be easily analyzed and optimized by algorithms?" Alternatively, since visualizations are a big part of exploratory data analysis, what is a good way to design a grammar that makes it work more directly with data analysis operations?

3.1.4 Appropriate levels of abstraction. Given that the visualization grammars are often implemented on top of lower-level graphics libraries, it would be interesting to think about how high-level grammars and low-level specifications can work together. As an example, Vega-Lite [14], a high-level grammar, compiles down to lower-level Vega [15] specifications. An interoperable stack of visualization languages on different abstraction levels, such as the D3—Vega—Vega-Lite—Voyager stack [2, 14, 15, 22], can be useful for code reuse and moving across languages. What is the appropriate level of encapsulation for which user group, and what vocabulary will be understandable by whom? It is conceivable that a highly encapsulated grammar can resemble chart-type based specifications, but with GoG-based defaults set under the hood. Since one advantage of GoG is explorability [7], is there a trade-off between encapsulation and explorability, *i.e.*, if the designer encapsulates the grammar with a chart type like "bar chart", does it lead to less exploration of alternative visualization designs?

3.2 Better evaluation for visualization grammars

Currently, if we are judging a grammar based on criteria such as expressivity, we can make an example gallery and use expert evaluation to make our case. These methods assess the usefulness of the grammars at a low cost, but they are subjective and do not tell us much about how users interact with the grammar. What are ways to better evaluate visualization grammars that capture how the grammars are used and make sure that the grammars serve their *purpose*?

3.2.1 Traditional user studies for visualization grammars. Is there value in conducting user studies on visualization grammars? User studies might be difficult to recruit for, take longer time, or the participants can be too unfamiliar (or familiar) with the grammar. A GoG-based grammar can be a more convoluted and error-prone way of specification compared to chart-type based libraries if the user does not understand and takes advantage of its flexibility. In literature, there is suggestion that learning a visualization specification tool can be helped by Voyager 2, a mix-initiative recommender system [22], and Vega-Lite has been used as a design prototype to understand K-12 teachers' preferences for visualization tools [10]. We would like to see more academic research and experiences from industry in understanding how well people learn, understand and take advantage of the Grammar of Graphics abstraction.

3.2.2 Alternative evaluation methods? Given the weakness of standard user studies when applied to visualization grammars, can there be better-suited evaluation questions and methodologies? There can be a mismatch between what we can answer from traditional user studies in the CHI community and what we want visualization grammars to achieve. How can we translate the criteria for a "good" visualization grammar into concrete evaluation questions?

3.2.3 Learning from real-world usage. Since there are several popular GoG implementations in the wild [14, 17, 19], what can we learn

[4] https://vega.github.io/vega/
[5] https://ggplot2.tidyverse.org/reference/index.html

from their real-world usages? Visualization grammar users might take full advantage of the expressiveness and explorability of GoG, or they could simply copy and paste from example code. What can we expect to learn from usage patterns and how can those patterns inform future grammar designs? If the users are not leveraging the grammar abstractions, how can we help the users achieve what they want and even expand their capacity, such as discovering or deducing a visualization specification?

4 ORGANIZATION OF THE SIG

We welcome researchers, students, and practitioners from communities such as CHI, VIS, UIST, CSCW, and KDD. We expect that participants with different levels of experience in visualization grammars can all contribute. Even though some participants may not have experience developing a visualization grammar, they can have created visualizations or have domain expertise in data work and evaluation methods.

We plan to organize the SIG into three parts. The first part will be brief presentations by the organizers including self-introductions, background on visualization grammars, and a list of prepared discussion topics. Then, we will solicit additional experiences and topics from participants, forming discussion points for the third part, the breakout group. The discussion points can evolve around *what makes a grammar "good"* and *better evaluation methods for visualization grammars*. Since the SIG will be virtual, we will use video-conferencing features such as chat and break-out rooms.

5 CONCLUSION

The goal of this SIG is to start a conversation around hard questions in designing and evaluating visualization grammars. This SIG can bring together researchers and practitioners in the visualization and broader HCI community. In addition to drawing on lessons learned from developing existing visualization grammars, we want to solicit new ideas in defining what makes a grammar "good" and reflecting on evaluation methods for grammars. Our conversation can cover refining the abstractions in visualization grammars, handling the combinatorics complexity in grammar specifications, enabling a visualization ecosystem supporting multiple abstraction levels and tasks, envisioning evaluation methods that capture purposes of grammars, and measuring how visualization grammars are used in the wild.

REFERENCES

[1] A. F. Blackwell, C. Britton, A. Cox, T. R. G. Green, C. Gurr, G. Kadoda, M. S. Kutar, M. Loomes, C. L. Nehaniv, M. Petre, C. Roast, C. Roe, A. Wong, and R. M. Young. 2001. Cognitive Dimensions of Notations: Design Tools for Cognitive Technology. In *Cognitive Technology: Instruments of Mind*, Meurig Beynon, Chrystopher L. Nehaniv, and Kerstin Dautenhahn (Eds.). Springer Berlin Heidelberg, -, 325–341.

[2] M. Bostock, V. Ogievetsky, and J. Heer. 2011. D^3 Data-Driven Documents. *IEEE Transactions on Visualization and Computer Graphics* 17, 12 (Dec. 2011), 2301–2309. https://doi.org/10.1109/TVCG.2011.185

[3] T. Ge, Y. Zhao, B. Lee, D. Ren, B. Chen, and Y. Wang. 2020. Canis: A High-Level Language for Data-Driven Chart Animations. *Computer Graphics Forum* 39, 3 (June 2020), 607–617. https://doi.org/10.1111/cgf.14005

[4] Kieran Healy. 2018. *Data visualization: a practical introduction.* Princeton University Press, -.

[5] BBC Visual and Data Journalism. 2019. How the BBC Visual and Data Journalism team works with graphics in R. https://medium.com/bbc-visual-and-data-journalism/how-the-bbc-visual-and-data-journalism-team-works-with-graphics-in-r-ed0b35693535

[6] Y. Kim and J. Heer. 2020. Gemini: A Grammar and Recommender System for Animated Transitions in Statistical Graphics. *IEEE Transactions on Visualization and Computer Graphics* 27, 02 (2020), 1–1. https://doi.org/10.1109/TVCG.2020.3030360 Conference Name: IEEE Transactions on Visualization and Computer Graphics.

[7] Younghoon Kim, Kanit Wongsuphasawat, Jessica Hullman, and Jeffrey Heer. 2017. GraphScape: A Model for Automated Reasoning About Visualization Similarity and Sequencing. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 2628–2638. https://doi.org/10.1145/3025453.3025866 event-place: Denver, Colorado, USA.

[8] Jianping Kelvin Li and Kwan-Liu Ma. 2020. P6: A Declarative Language for Integrating Machine Learning in Visual Analytics. *IEEE Transactions on Visualization and Computer Graphics* 27, 02 (2020), 1–1. https://doi.org/10.1109/TVCG.2020.3030453

[9] Jock Mackinlay. 1986. Automating the design of graphical presentations of relational information. *ACM Transactions on Graphics (TOG)* 5, 2 (April 1986), 110–141. https://doi.org/10.1145/22949.22950

[10] Bahare Naimipour, Mark Guzdial, and Tamara Shreiner. 2020. Engaging Pre-Service Teachers in Front-End Design: Developing Technology for a Social Studies Classroom. In *2020 IEEE Frontiers in Education Conference (FIE)*. IEEE, Uppsala, 1–9. https://doi.org/10.1109/FIE44824.2020.9273908

[11] Deokgun Park, Steven Mark Drucker, Roland Fernandez, and Niklas Elmqvist. 2017. ATOM: A Grammar for Unit Visualizations. *IEEE Transactions on Visualization and Computer Graphics* 24, 12 (2017), 3032–3043. https://doi.org/10.1109/TVCG.2017.2785807

[12] plotly. 2019. Introducing Plotly Express. https://medium.com/plotly/introducing-plotly-express-808df010143d

[13] Xiaoying Pu and Matthew Kay. 2020. A Probabilistic Grammar of Graphics. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. ACM, Honolulu HI USA, 1–13. https://doi.org/10.1145/3313831.3376466

[14] Arvind Satyanarayan, Dominik Moritz, Kanit Wongsuphasawat, and Jeffrey Heer. 2017. Vega-Lite: A Grammar of Interactive Graphics. *IEEE Transactions on Visualization and Computer Graphics* 23, 1 (2017), 341–350. https://doi.org/10.1109/TVCG.2016.2599030

[15] Arvind Satyanarayan, Ryan Russell, Jane Hoffswell, and Jeffrey Heer. 2016. Reactive Vega: A Streaming Dataflow Architecture for Declarative Interactive Visualization. *IEEE Transactions on Visualization and Computer Graphics* 22, 1 (Jan. 2016), 659–668. https://doi.org/10.1109/TVCG.2015.2467091

[16] C. Stolte, D. Tang, and P. Hanrahan. 2002. Polaris: a system for query, analysis, and visualization of multidimensional relational databases. *IEEE Transactions on Visualization and Computer Graphics* 8, 1 (Jan. 2002), 52–65. https://doi.org/10.1109/2945.981851

[17] Jake Vanderplas and Brian Granger. 2020. altair-viz/altair. https://github.com/altair-viz/altair original-date: 2015-09-19T03:14:04Z.

[18] Hadley Wickham. 2010. A Layered grammar of graphics. *Journal of Computational and Graphical Statistics* 19, 1 (2010), 3–28. https://doi.org/10.1198/jcgs.2009.07098 ISBN: 1061-8600.

[19] Hadley Wickham. 2016. *ggplot2: elegant graphics for data analysis.* springer, -.

[20] Leland Wilkinson. 2005. *The Grammar of Graphics.* Springer-Verlag, New York. http://link.springer.com/10.1007/0-387-28695-0

[21] Krist Wongsuphasawat. 2020. Encodable: Configurable Grammar for Visualization Components. In *VIS (InfoVis/VAST/SciVis) 2020*. IEEE, -, 5. http://arxiv.org/abs/2009.00722 arXiv: 2009.00722.

[22] Kanit Wongsuphasawat, Zening Qu, Dominik Moritz, Riley Chang, Felix Ouk, Anushka Anand, Jock Mackinlay, Bill Howe, and Jeffrey Heer. 2017. Voyager 2: Augmenting Visual Analysis with Partial View Specifications. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. Association for Computing Machinery, New York, NY, USA, 2648–2659. https://doi.org/10.1145/3025453.3025768