GraphTrials: Visual Proofs of Graph Properties

Henry Förster
Department of Computer Science,
University of Tübingen, Germany

Felix Klesen
Institute of Computer Science,
University of Würzburg, Germany

Tim Dwyer
Faculty of Information Technology,
Monash University, Australia

Peter Eades
School of Computer Science,
The University of Sydney, Australia

Seok-Hee Hong
School of Computer Science,
The University of Sydney, Australia

Stephen G. Kobourov
Department of Computer Science,
University of Arizona, Tucson, AZ, USA

Giuseppe Liotta
Department of Engineering,
University of Perugia, Italy

Kazuo Misue
Department of Computer Science,
University of Tsukuba, Japan

Fabrizio Montecchiani
Department of Engineering,
University of Perugia, Italy

Alexander Pastukhov
Department of Psychology,
University of Bamberg, Germany

Falk Schreiber
Department of Computer Science,
University of Konstanz, Germany

Abstract

Graph and network visualization supports exploration, analysis and communication of relational data arising in many domains: from biological and social networks, to transportation and powergrid systems. With the arrival of AI-based question-answering tools, issues of trustworthiness and explainability of generated answers motivate a greater role for visualization. In the context of graphs, we see the need for visualizations that can convince a critical audience that an assertion about the graph under analysis is valid. The requirements for such representations that convey precisely one specific graph property are quite different from standard network visualization criteria which optimize general aesthetics and readability.

In this paper, we aim to provide a comprehensive introduction to visual proofs of graph properties and a foundation for further research in the area. We present a framework that defines what it means to visually prove a graph property. In the process, we introduce the notion of a visual certificate, that is, a specialized faithful graph visualization that leverages the viewer's perception, in particular, pre-attentive processing (e. g. via pop-out effects), to verify a given assertion about the represented graph. We also discuss the relationships between visual complexity, cognitive load and complexity theory, and propose a classification based on visual proof complexity. Finally, we provide examples of visual certificates for problems in different visual proof complexity classes.

2012 ACM Subject Classification Human-centered computing → Visualization theory, concepts and paradigms; Human-centered computing → Graph drawings; Human-centered computing → Information visualization

Keywords and phrases Graph Visualization, Theory of Visualization, Visual Proof

Digital Object Identifier 10.4230/LIPIcs.GD.2024.16

Related Version *Full Version*: `https://arxiv.org/abs/2409.02907`

Acknowledgements This work was initiated at Dagstuhl seminar 23051 "Perception in Network Visualization". We thank the organizers for making this fruitful interdisciplinary exchange possible and all participants for interesting discussions and insights during the seminar week.

© Henry Förster, Felix Klesen, Tim Dwyer, Peter Eades, Seok-Hee Hong, Stephen G. Kobourov, Giuseppe Liotta, Kazuo Misue, Fabrizio Montecchiani, Alexander Pastukhov, and Falk Schreiber; licensed under Creative Commons License CC-BY 4.0

32nd International Symposium on Graph Drawing and Network Visualization (GD 2024).
Editors: Stefan Felsner and Karsten Klein; Article No. 16; pp. 16:1–16:18
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

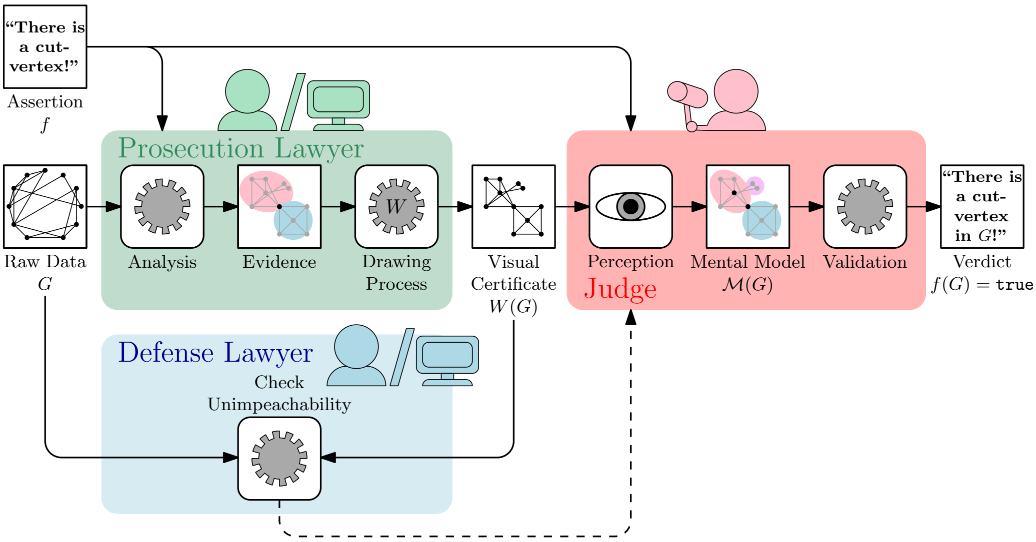


Figure 1 Our model GRAPHTRIALS identifies key processes for visually proving an assertion about a given graph in an adversarial setting. The *prosecution lawyer*, i. e., a software or a human (assisted by software), intends to highlight evidence for a graph being guilty of satisfying an assertion using a visual certificate drawing. To convince the *judge*, i. e., the human audience of the drawing, the visual certificate guides the judge's perception to form a mental model which makes the assertion easy to validate. The visual certificate must be unimpeachable as a *defense lawyer* (software or human adversary) checks for reasons to doubt the certificate's validity to influence the judge's verdict.

1 Introduction

While state-of-the-art graph and network visualization techniques do a reasonable job of untangling graphs to convey meaning and support free-form exploration, there are certain application scenarios where these algorithms fall short. Namely, we focus on applications where it is necessary to *convince* a (possibly non-expert) audience that a particular graph has some structural property. We emphasize that this kind of application scenario differs significantly from the traditional usage of visualization to generate new knowledge. Namely, existing graph and network visualization techniques have sought mainly to represent all aspects of a graph or network structure as faithfully as possible such that a user can explore the visualization, identify structures, and gain insights about the underlying data. These traditional visualization techniques can be sufficient for journalists and other communicators to support a narrative in print or on-line media [11] by showing only selected views of graphs.

However, novel approaches are required in our setting in which a specific property of the data is to be conveyed in an adversarial setting where the validity of the evidence presented may be questioned (see also the defense lawyer role in Fig. 1 which may, e. g., represent doubts of the audience). For example, the investigative activity of the Italian Revenue Agency (IRA) exploits the visual analysis of social networks whose nodes are the actors of potential fraudulent activities and whose edges represent financial/legal transactions between the actors. The investigators of IRA who suspect a group of persons or a single individual/company of tax evasion submit a case to the Italian financial Police for possible prosecution, which also implies showing some structural properties of the network beyond reasonable doubts. See, e. g., [21–23] for references about the use of visual analytics in the context of contrasting tax evasion in Italy.[1] Below we describe introductory examples.

[1] One author has been approached by the Australian Security and Investments Commission (a governmental regulator for stock exchange) inquiring about visualizations to convince a court about illegal trades.

▶ **Example 1.** A network admin discovers that two critical parts of the infrastructure would not be able to communicate with each other if a particular switch fails. To increase the robustness of the network, new hardware is needed. They have to convince the manager, who has no background in network security, to fund new hardware.

▶ **Example 2.** In a legal court case, the prosecution discovered that money acquired in black market sales was laundered by laundromat chain as evidenced by money provably transferred via a complicated network from the dealers to the laundromats. The prosecution has to convince the judge that all suspects belong to the criminal syndicate.

▶ **Example 3.** A new AI based heuristic is able to efficiently decide if a given graph is Hamiltonian, i. e., to test if it contains a cycle traversing all its vertices exactly once[2]. However, false positives must be filtered out. A human operator needs to perform this task as there is no efficient algorithm. To facilitate this, the new version of the algorithm should also create a visualization of the graph making the Hamiltonian cycle obvious to the operator.

Such scenarios have key differences to standard motivations for graph visualization. Typical graph visualization techniques (node-link layout algorithms [18, 46], matrix ordering approaches [7] and mixed approaches which either include features of different paradigms [3, 4, 29] or show different visualizations side-by-side [13, 28]) usually seek a representation showing as many graph properties as possible simultaneously (by trading off aesthetic and readability criteria [1, 8, 9, 17]). However, for the scenarios above it is better to focus on showing optimally and faithfully just one specific property, i. e., we want a *visual proof* for that property.

More precisely, a visual proof is a proof given by the use of a graphical or visual representation called *visual certificate*. A good visual proof should be clear and concise, conveying the main idea in an easy-to-understand way. It should be able to effectively communicate the desired message without being overly complex or cluttered. Additionally, the visual certificate should be aesthetically pleasing and easy to interpret. Somehow, it should be able to provide evidence to support the argument being made. Thus, a good visual certificate should be accurate, concise, and free of errors or mistakes.

In fact, visual proofs are already used in mathematics and other areas such as logic, graph theory, computer science, and physics [37, 57]; visual proofs are often easier to understand than algebraic proofs, as they are less abstract and easier to follow. Accessible proofs are often considered more beautiful by mathematicians; e. g., Appel and Haken employed a computer-assisted proof of the long-open four-color theorem in 1976 [5]. This new type of proof sparked philosophical debates [50] and while the theorem is broadly accepted as proved[3], researchers still desire a more elegant proof [2]. Thus, we expect that visual proofs are appealing and even more convincing to experts also in fields other than mathematics.

Visual proofs can also convey properties to non-expert users or explain correctness of AI-generated solutions. As powerful chat-based interfaces are capable of generating plausible sounding – but difficult to verify – explanations of complex phenomena, we believe that there is a requirement to understand what makes a graph representation a proper visual certificate.

[2] Note that neural network approaches for NP-hard problems have been described, e. g., in [10]. In addition, the need for visualizations in the context of explainable deep learning has been described, e. g., in [16].
[3] According to the *Oxford English Dictionary*, it is yet to be proven as a "*mathematical* theorem" [40].

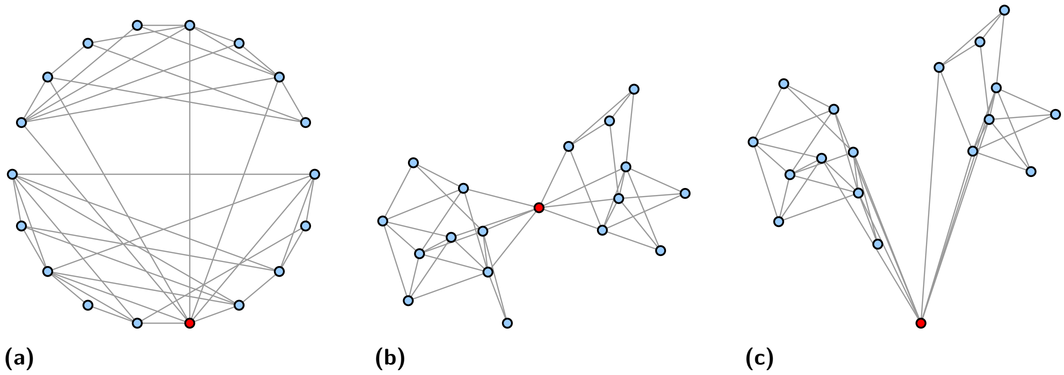


Figure 2 Three layouts generated with yEd[4] [59]. (a) and (b) Circular and organic layouts generated with standard settings, resp. (c) Manually created layout highlighting the cut-vertex.

Contribution. We introduce a model identifying important steps and their interactions in a visual proof of a graph property. Based on this model, we formalize the concept of visual certificates and give requirements for a visualization to qualify as such. We also give examples of visual proofs for widely used graph properties and identify open research questions that should be answered to better understand visual proofs and make them algorithmically usable.

2 First Examples of Visual Proofs

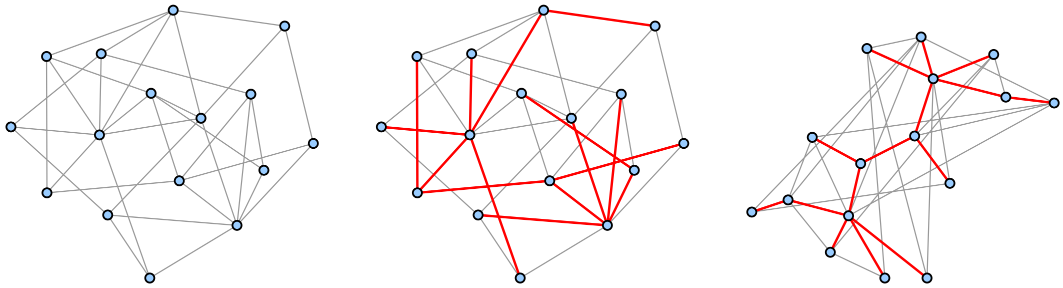
2.1 Example 1: The Graph contains a Cut-Vertex

First, we revisit Example 1. In this communication network there are two distinct parts such that all connections between them traverse a single switch. This corresponds to the graph underlying the network containing a *cut-vertex*, whose removal separates the remainder of the graph into at least two distinct components. Hence, in order to convince the manager, the network admin has to point out that the graph underlying the network can be separated by the removal of the vertex corresponding to the switch. So, they first layout the graph using a circular layout, which is a wide-spread all-purpose visualization style [44], and point the manager to the fact that the red colored vertex is a cut-vertex; see Figure 2a.

Unfortunately, the circular layout does a poor job at highlighting the cut-vertex. While it is evident to the manager that there are a top and a bottom component connected by some edges, they explain that they are not sure if all connections between both components use the suggested cut-vertex or not. Hence, the network admin prepares a second drawing using a force-directed organic layout where the cut-vertex is clearly visible; see Figure 2b. However, the engineer who designed the network becomes defensive and claims that there could be another edge hidden behind the alleged cut-vertex. This argument can be easily disproven by the network admin as they move the cut-vertex down, obtaining the drawing in Figure 2c. Presented with this new line of evidence, the engineer stops arguing and the manager agrees that the network has to be made more robust.

Discussion. This example illustrates how standard layout techniques may be unable to highlight even simple properties. In the circular layout, it is not easy to verify even when the cut-vertex is highlighted; see Figure 2a. This is due to the Gestalt principle of grouping [52,53].

[4] Unless specified otherwise, the layouts of all visualizations in this paper have been created by the authors.



(a) (b) (c)

Figure 3 (a)–(b) Organic layout generated with standard settings by yEd [59], with a spanning tree highlighted in (b). (c) A manually created layout highlighting the spanning tree.

Here, the initial perception is guided by continuity and closure of node positions, leading to the perception of a single circular component. As a second step, an observer may see two separate components with edges biasing perception due to connectedness grouping. Thus, the observer has to analyze the entire graph, going node-by-node, to negate the automatic perceptual grouping induced by the layout to verify that there is a cut-vertex. The issue with the second illustration in Figure 2b is of different nature. Namely, the force-directed layout does a much better job at highlighting the cut-vertex. In fact, the observer discovers two dense salient features which are the two components separated by the cut-vertex and immediately notes that they are connected at a single vertex. Nevertheless, if there is an overlapping edge behind the cut-vertex, the drawing may look the same, challenging the human observer to identify that the vertex is not a cut-vertex. The drawing in Figure 2c avoids this problem by explicitly highlighting the cut-vertex via pre-attentively perceptable patterns (i. e., pop-out effects) [53]. The singular goal of highlighting the cut-vertex is achieved at the cost of traditionally accepted aesthetic metrics [42], as – compared to the circular and force-directed layouts – the general layout is unbalanced, with many crossings and poor resolution; see Table 1. Thus, visual certificates may not be useful in traditional exploratory applications, instead they focus on highlighting a specific property.

We remark that a cut-vertex proves non-2-connectivity and a similar approach can be used to visually prove that a graph is *not k-connected*: there exists a set of $k-1$ vertices whose removal separates the graph and we can layout the graph so that all connections between two clearly separated parts run via this vertex set.

2.2 Example 2: The Graph is Connected

In Example 2, to convince the judge, the prosecution lawyer decides to visualize the network of criminals induced by the connections of provable money transfers. The prosecution lawyer draws it with a force-directed approach; see Figure 3a. While Figure 3a shows that there are many connections in the graph, it does not emphasize that there is only a single connected component. Hence, the defense lawyer argues that the component containing their client may have been drawn on top of the component with all the convicted criminals. Hence, the prosecution lawyer has to improve their visual proof. To do so, they include a highlighted *spanning tree* that shows that every vertex can be reached from every other vertex; see Figure 3b. Although the defense lawyer now has to admit that there is a smaller portion of the drawing to check, i. e., the highlighted edges, their argument stays more or less the same: that there are still crossings between edges of the spanning tree, which may be due

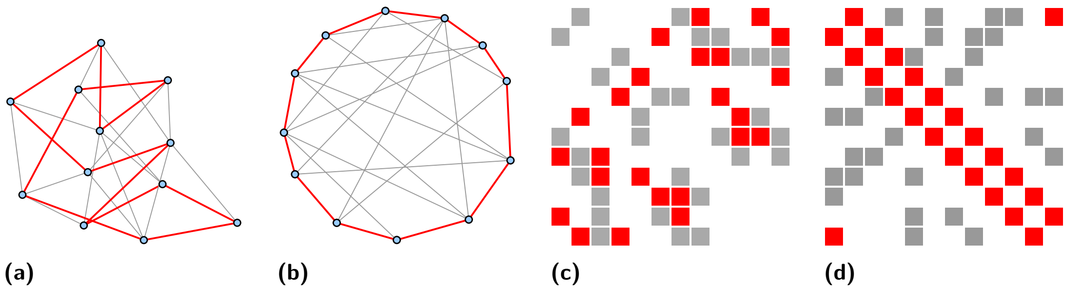


Figure 4 Four layouts of a graph with a Hamiltonian cycle (red).

to two different highlighted components drawn on top of each other. Thus, the prosecution lawyer creates a third drawing in which the spanning tree is crossing-free; see Figure 3c. Here the spanning tree is rooted at the central vertex and vertices are drawn on concentric circles depending on distance from the root. Given this visualization, the defense rests, and the judge decides quickly that indeed all members of the network are affiliated.

Discussion. While in Example 1 we have seen that the drawing style of the entire graph can be important to visually prove a property, here we added another dimension. Namely, a subgraph is explicitly color-highlighted for pre-attentive perception. In addition, the drawing of this subgraph was very important in creating a convincing argument. In Figure 3b the drawing of the spanning tree is not very readable. Thus, even with the attention drawn to this portion of the drawing, it remains time consuming to check that a single tree connects all vertices. But when the tree is laid out in a concise and readable fashion as in Figure 3c, it is quite evident that it spans all the vertices, as the colored edges induce automatic grouping via similarity [52] and act as guidance for attention spread [30]. Similar to Example 1, while the quality of the drawing of the spanning tree is improved, the drawing of the rest of the graph does not measure well on the usual metrics; see Table 1.

2.3 Example 3: The Graph has a Hamiltonian Cycle

We may train an AI to produce a good node-link drawing for Example 3. As in Section 2.2, we observe that the quality of the drawing of the evidence relevant to the property under consideration is more important than the drawing of the full graph (Figure 4a), thus we select a circular layout with the hamiltonian cycle forming the outer face (Figure 4b). However, the human operator needs to check that all edges of the highlighted outer cycle are indeed present which can become increasingly difficult for larger graphs where resolution may become problematic. We can improve upon these issues by instead using an adjacency matrix representation. While an arbitrary permutation (see Figure 4c) does not provide any insights, an appropriate sorting of rows and columns makes the cycle composed of three components: one red diagonal and two red-cells (top-right and bottom-left); see Figure 4d.

Discussion. We observed that different visualization paradigms may perform better or worse for visually proving a property. While the node-link drawing in Figure 4b already highlights the cycle well, the adjacency matrix representation in Figure 4d composes the Hamilton cycle in three components. The perception of the red diagonal is facilitated by figure-ground separation via connectedness and similarity [52], and the two corner cells stand out due to both color difference and symmetry [58]. The particular advantage of this representation is

Table 1 Selected aesthetic metrics of the node-link drawings in this paper: stress ST [32], node resolution NR, Jaccard index JI [1], edge-length ratio EL [33,34], crossing resolution CR [19,20], crossing number CN, aspect ratio AR and angular resolution AN [39]. Red numbers in parantheses give the corresponding values for subgraphs highlighted in red in the corresponding figure. For ST and CN lower numbers are better, otherwise higher numbers are better.

Fig.	2			3		4		5	6			7	
	(a)	(b)	(c)	(a) – (b)	(c)	(a)	(b)	(d)	(a)	(b)	(c)	(a)	(b)
ST	132.1	9.2	36.7	13.6 (31.3)	32.5 (19.6)	9.3 (51.9)	45.5 (12.6)	6635	567.6 (9.1)	651.0 (1.9)	1022.7 (3.3)	58.8	94.4
CN	81	19	26	28 (8)	63 (0)	29 (4)	45 (0)	7452	6211 (2)	9493 (0)	9992 (0)	12649	12650
JI	.274	.332	.283	.330 (.132)	.285 (.160)	.407 (.167)	.339 (.232)	.037	.177 (.243)	.180 (.361)	.182 (.361)	.918	.920
EL	.174	.333	.127	.360 (.404)	.149 (.382)	.401 (.649)	.185 (.542)	.090	.020 (.161)	.023 (.586)	.013 (.472)	.126	.047
NR	.174	.104	.100	.181 (.181)	.119 (.119)	.164 (.164)	.184 (.184)	.020	.016 (.123)	.016 (.314)	.011 (.307)	.126	.047
AR	.985	.774	.966	.884 (.884)	.796 (.796)	.871 (.871)	.987 (.987)	1	.941 (.606)	.941 (.977)	.994 (.994)	.998	.898
CR	20.0	27.1	7.2	37.1 (37.1)	20.5 (N/A)	31.4 (48.2)	22.5 (N/A)	2.6	0.79 (73.2)	0.79 (N/A)	0.74 (N/A)	14.4	4.68
AN	10.0	.20	.56	1.68 (20.0)	4.44 (21.4)	0.40 (15.93)	12.08 (141.0)	0.21	.017 (6.87)	.017 (112.0)	.007 (97.6)	7.20	1.23

that the used visual cues scale nicely even for very large matrices (up to the pixel resolution of the screen) [58]. Thus, an important criterion for judging the quality of a visual proof should be the workload required by the observer to evaluate the correctness. As checking for Hamiltonicity is a difficult task with an all-purpose visualization (see also Section 3.3), both visualizations should be regarded as valid visual certificates albeit of different quality.

3 Related Theories, Frameworks and Models

3.1 Certifying Algorithms

The concept of visual certificates is related to *certifying algorithms* popularized by McConnell et al. [36], which seek to provide short and easy-to-check *certificates* for the correctness of an algorithm. Let $f : X \to Y$ be a computable, surjective function for input set X and output set Y and let W be a set of *witnesses*. Intuitively speaking, a witness describes a simple proof certifying that the output y of an algorithm for f on input x satisfies $f(x) = y$. The validity of a witness for a certain combination of inputs and outputs is assessed via the witness predicate $\mathcal{W} : X \times Y \times W \to \{\texttt{true}, \texttt{false}\}$ that fulfills:

1. *Witness property:* Given $(x, y, w) \in X \times Y \times W$, it holds $f(x) = y \Leftrightarrow \mathcal{W}(x, y, w) = \texttt{true}$.
2. *Checkability:* Given $(x, y, w) \in X \times Y \times W$, it is trivial to determine $\mathcal{W}(x, y, w)$.
3. *Simplicity:* $\mathcal{W}(x, y, w) \Rightarrow f(x) = y$ has a simple proof.

An algorithm for f is now called *certifying algorithm* if for any input $x \in X$ it computes the output $y = f(x) \in Y$ and a witness $w \in W$ such that $\mathcal{W}(x, y, w) = \texttt{true}$. It is worth noting that Properties 2 and 3 of the witness predicate are vaguely formulated. McConnell et al. [36] suggest that Property 2 can be formalized by requiring that there must be a decision algorithm for $\mathcal{W}$ that runs in a certain time (such an algorithm is called a *checker*). On the other hand, they emphasize that Property 3 is intentionally left subjective as it relies on what is considered common knowledge. For examples of certifying algorithms, see the full version of this article.

3.2 Perception

Visual proofs are concerned with a design of visual evidence for an existence of a specific property, such as the presence of a cut-vertex, a Hamiltonian cycle, etc. In principle, a proof for such a property can be reduced to a program that returns a binary outcome, affirming or rejecting the claim. This may be sufficient for specialists who are familiar with the property itself, understand and trust the algorithm behind the code, and trust that the code is valid. However, such evidence may not be convincing to a non-specialist (a judge, a stockholder, etc.), particularly because the proof itself will be just one piece of evidence among many. Prior research shows that in such cases, presenting evidence *per se* is not enough, as information can be discounted as confusing, unimportant, or, given the wrong context, even misleading [49], as the accessibility and clarity of evidence could be as important as evidence itself [27].

Due to the diversity of graph properties there can be no general solution. Visual proof design might be guided by the principle of optimizing the *data-ink* ratio [49]. Thus, instead of optimizing overall aesthetic metrics [42], one should minimize the required number of *visual queries*, i. e., attention orientation, driving eye movements, and pattern/object recognition [54].

The human visual perception system consists of three stages: (1) rapid parallel processing involving billions of neurons, e. g., extraction of orientation, texture, color, and motion features; (2) slower processing than Stage 1, e. g., detection of 2D patterns, contours and regions; (3) slow serial processing, involving both working and long-term memory, e. g., object identification [53]. As in Stage 1 the entire visual field is processed quickly in parallel, information that can be captured in this stage can be easily distinguished. Thus, pre-attentive (pop-out) patterns such as color, size, orientation, shapes, etc. should be utilized.

In other words, a good visual proof must ensure that a focal piece of evidence is a visual "pop-out" feature that automatically attracts viewer attention and that the visual layout is parsed and grouped into patterns that express the evidence. In case of the former, studies on visual search provide a comprehensive list of useful pop-out features such as color, size, contrast, or location [58]. Regarding the latter, one can rely on a large body of literature on principles of perceptual organization, commonly known as Gestalt principles [52]. However, yet another constraint is placed by our working memory that limits the number of nodes, edges, and components that can realistically be assessed at any single time [35].

The examples above illustrate the importance of this approach for visual proofs. For instance, consider the visual evidence for the existence of a cut-vertex in Figure 2a. While it uses color to attract the viewer's attention to the cut-vertex and spatial arrangement to visually separate the two components, it still leads to an excessive number of visual queries, requiring multiple scans of individual vertices to ensure that they are connected only to the cut-vertex and the nodes within the component. In turn, there is a memory bottleneck that is likely to prevent a viewer from being completely certain about the validity of the proof. In contrast, in Figure 2c the graph layout groups the entire evidence in just three components and clearly shows lack of inter-component edges, so that very few visual queries are required to confirm the vertex is indeed a sole connector between the components. In short, although there cannot be a single one-size-fits-all approach for constructing visual proofs, their critical role in aiding the cognition of the viewer means they should be built based on principles of perceptual organization and around the limitations of attention and memory [54].

3.3 Computational Complexity

To evaluate the amount of the cognitive workload, we will apply concepts from complexity theory [6, 24]. It is also worth mentioning that the examples discussed so far differ in terms of their computational complexity. Namely, all cut-vertices of a graph and a spanning-tree can be found in time $O(n + m)$ based on BFS traversals where n is the number of vertices and m the number of edges while determining a Hamiltonian cycle is *NP-complete* [24]. Thus, in Example 3, we have visually proven an algorithmically difficult to solve problem.

However, there may be graph properties that cannot be visually proven. We first have to discuss how a human observer interacts with a visual certificate. In Example 1, the human observer identified two connected components and then saw that they can be separated by the removal of their shared vertex. Such a procedure could be seen as an $O(1)$ time algorithm, where the observer determined that there is only a single point where both components touch. Similarly, in Examples 2 and 3, the observer may have checked for every vertex if it was part of the highlighted structure. Even if they were to check this for every vertex one at a time, the resulting algorithm would still run in linear time. Hence, an observer is actually performing a *deterministic* validation algorithm for establishing that a certificate is correct.

Now, consider the complementary question to Example 3, i. e., we want to determine whether a graph does *not* contain a Hamiltonian cycle. This is a *CoNP-complete* problem as it is the complement to an NP-complete problem. For CoNP-complete problems it is likely that there is no certificate that can be checked in polynomial time [6], i. e., if we assume that a human observer deterministically analyzes a visualization (as could be recreated by computer vision), we have to assume that we cannot visually prove a CoNP-complete problem.

3.4 Related Visualization Models

Aside from graph visualizations, the concept of visually enhancing a proof is wide-spread. In mathematics, visual proofs for theorems have been used since ancient times [57] and there is a plethora of examples [37]. The question if such proofs can be regarded as such also has been discussed philosophically [12]. Also in computer science, visualizations are heavily used to convey knowledge, e. g., while not necessarily proving, an interactive sequential art by Bret Victor [51] beautifully explained an algorithm from a Nature paper [55].

Overall, there is a trend of increasingly sophisticated models considering an holistic integration of visualization into the sensemaking process, typically with the goal of informing the design of interactive systems for data exploration. Early models considered a linear pipeline, from data, via various transformations, to a visual display [14]. Visual analytics seeks to apply visualization to support the entire human sense-making loop [41]. More recent models aim to connect sense-making from interactive data visualization, via hypothesis formation and testing, to knowledge generation [43]. An underlying theme across most of this work is the role of computational guidance in the analytics process, and how algorithms can support the various loops in the sensemaking process [15]. By contrast, we consider a different model to conceptualize the role of algorithms, and AI, in supporting data (specifically network data) understanding. Our model for visual proofs (Fig. 1) does not seek to replace the traditional sense-making/knowledge-generation loop, but to support humans in situations where the result of a complex algorithm or property needs to be explained and justified.

There are also models related to ours from information visualisation research. Song et al. [45] considered a problem that may be seen as a complementary question to the one studied in this paper: They investigated how computer vision can understand network visualizations optimized for human users. Wickham et al. [56] proposed a two-phase procedure to convince a human observer that a data set contains statistically significant difference from randomly generated data. The human observer is first exposed to several randomly generated data sets (similar to a Rorschach test) before being exposed to a line-up consisting of the real data set and a couple randomly generated data sets. The first phase primes the human viewer for statistically insignificant variations so that, in the second phase, statistically significant differences clearly pop out from the noise. Another related model are *Gragnostics*, which are ten features suggested by Gove [25,26], that are fast to compute and provide a quantification of structural graph properties. In contrast to our model that aims to prove structural properties of graphs, Gragnostics provides the human user with a first impression of the structure of the graph at hand which may be helpful for initiating a thorough investigation. Finally, our model may also be seen as a visual communication of structural graph properties. Visual communication has been investigated in other settings for several decades, see e. g. [47, 48].

4 The GraphTrials Model

We are now ready to discuss our formalization of *visual proofs*. For this, we first abstractly outline the process of visually proving properties of graphs in an adversarial setting using a model that we call GraphTrials; see also Figure 1. The model includes three distinct roles that have already appeared in our discussion of Example 2 in Section 2.2: The *prosecution lawyer* must convince the judge that a certain assertion regarding a graph is true, the *defense lawyer* may raise doubts about the validity of the prosecution lawyer's claims, and the *judge* will determine the truth of the assertion. The roles *prosecution lawyer*, *judge* and *defense lawyer* are to be seen as abstract descriptions of the different actors in the process; e. g., in Examples 1 and 3, the prosecution lawyers were the network admin and the AI based algorithm, respectively. The latter example further indicates that not all roles have to be assigned to a human. In fact, we only require that the judge corresponds to the human audience of the visual certificate whereas each lawyer may be either human, software or a human assisted by software. Moreover, as we have seen in Section 2.1, it can also occur that a critical audience can act as both the judge and defense lawyer roles simultaneously.

To convince the judge of a valid assertion f for the input graph G, the *prosecution lawyer* draws a visual certificate $W(G)$. To do so, they first analyze the raw data G to reveal evidence that proves the assertion f. The evidence is then embedded in $W(G)$: a visual representation of G that in some way emphasizes the evidence. Note that in the scope of our model we treat the analysis of the raw data and extraction of the evidence as a black box, i. e., we may assume that the prosecution lawyer already knows that the assertion f is true for the input graph G and may also be given the evidence as input. This allows us to *efficiently* visually prove algorithmically difficult assertions (such as the existence of a Hamiltonian cycle as in Section 2.3) and to ignore how the evidence is gathered (either algorithmically or by human interaction) in our model. The latter aspect also provides the possibility to separate the evidence gathering from the visualization process W, i. e., W could be a reusable program that embeds the evidence according to a specification[5].

The *defense lawyer* checks the *unimpeachability* of $W(G)$ as a visual representation of G certifying $f(G)$. Thus, they may question whether the graph represented in the visualization actually corresponds to the input and they may also raise concerns if $W(G)$ is not distinguishable from a slightly different non-certificate (e. g., in Section 2.1 we encountered the case where an edge may have been hidden making it invisible to the judge's perception).

The *judge*, the human audience of the visual certificate $W(G)$, will validate the claim $f(G)$ using $W(G)$. In this step, the visual certificate $W(G)$ must guide the judge's perception so that they are able to form a mental model $\mathcal{M}(G)$ that facilitates confirmation of the validity of the assertion $f(G)$. For instance, the guidance can be formed by a suitable choice of topology which leads the judge to identify clusters of the graph as distinct salient features (as in Section 2.1) or by adding additional features such as color to draw attention to certain parts of the graph (as in Section 2.2). We discuss the judge's mental model in the full version of this article.

It is noteworthy that aside from the input graph and the verdict of the judge, the only information shared by all three roles is the *visual certificate* $W(G)$. In particular, it is the only medium that can be used by the prosecution lawyer to communicate the gathered evidence to the judge, i. e., the evidence is hidden information only accessible by the prosecution lawyer. Similarly, the judge is not communicating its mental model $\mathcal{M}(G)$ to the prosecution or defense lawyer, yet as we discussed above both roles might want to *estimate* what the mental model will look like. Furthermore, the nature of the mental model plays an important role in the validation step performed by the judge. Namely, the cognitive load put on the judge in this step depends hugely on how *complex* $\mathcal{M}(G)$ is. Finally, the defense lawyer's checking for unimpeachability is a process that is independent of the judge and prosecution lawyer and for a *faithful and readable* visual certificate we demand that there is no reason for the defense lawyer to raise doubts to the judge. As a result, there are several properties that we require from a visualization in order to call it a visual certificate and it could occur that an assertion cannot be *visually proven* for every graph for which the assertion is true (for instance we discussed issues related to scalability in Section 2.3). To this end, we also state when we want to say that a certain assertion can be visually proven for arbitrary graphs.

[5] The examples in Section 2.3 and 5 both use visual certificates that highlight cycles.

Visual Certificates and Visual Provability

We give formal requirements inspired by the concept of certifying algorithms discussed in Section 3.1. Let $f : \mathcal{G} \to \{\texttt{true}, \texttt{false}\}$ be an *assertion function* for the set of graphs $\mathcal{G}$, i. e., for some graphs the assertion $f(G)$ is `true` while for others it is not. For instance, if f is the existence of a cut-vertex, some graphs do contain one ($f(G) = \texttt{true}$) while others do not ($f(G) = \texttt{false}$). Consider a graph G with $f(G) = \texttt{true}$ and let $W(G)$ be a visualization of G. We call $W(G)$ *visual certificate* for $f(G)$ if and only if the following hold:

1. *Unimpeachability:* We call $W(G)$ *unimpeachable*, if it satisfies the following two properties. First, $W(G)$ should provide *information faithfulness* [38], i. e., it displays the ground truth properties and structures in G. Second, $W(G)$ should provide *task readablility* [38], i. e., the judge can *perceive* enough information for validating the assertion.
2. *Checkability:* Given $W(G)$, it is trivial to decide that $f(G) = \texttt{true}$. In particular, this means that the judge's perception leads to the formation of a *mental model* $\mathcal{M}(G)$ that makes it possible for the judge to *efficiently* validate the assertion. The number of distinct observations made by the judge in the process is called the *perceptual complexity*.
3. *Simplicity:* Given $\mathcal{M}(G)$, there is a *simple formal proof* for $f(G) = \texttt{true}$ that relies solely on conclusions that the judge may deduce using $\mathcal{M}(G)$. In particular, this means that $W(G)$ is *perceptually distinguishable* from any possible wrong visual certificate $W'(G)$.

If a visual certificate $W(G)$ exists for each $G \in \mathcal{G}$ with $f(G) = \texttt{true}$, we call f *visually provable*. Note that the complementary function f^c (which is `true` if and only if $f(G) = \texttt{false}$) needs not necessarily be visually provable. For instance, we were able to visually prove the assertion that G contains a Hamiltonian cycle in Section 2.3 but we argued that the absence of such a cycle cannot be visually proven in Section 3.3. This and requiring unimpeachability are clear differences to the concept of certifying algorithms whereas checkability and simplicity occur in both models, here considering the perceptual abilities of the judge; see also Section 3.1.

We are also interested in *how efficiently* the judge is able to validate $f(G) = \texttt{true}$ based on $\mathcal{M}(G)$. To this end, we define the *perceptual complexity* as the time that the judge needs to check the assertion given $\mathcal{M}(G)$. The perceptual complexity may depend on the size of the graph, however, in some scenarios (e.g. Example 1) it may be independent of it. Since we assume the judge to make an objective judgment based on the evidence, we can treat the thought process as a deterministic algorithm and apply methods from complexity theory to evaluate the perceptual complexity. See the full version of this article for an application of these concepts.

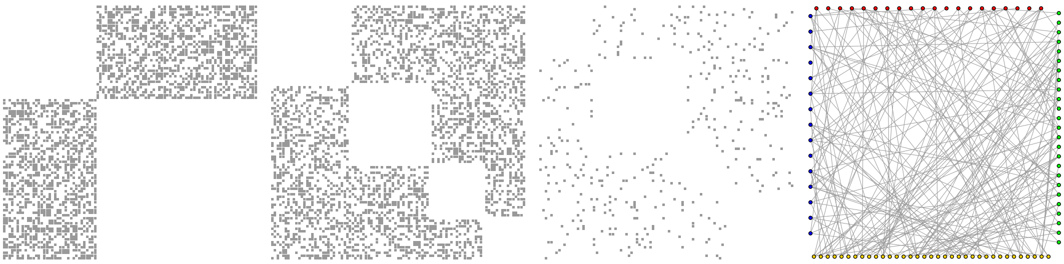
5 Visual Proofs for Graph Properties

We provide visual proofs for further widely used assertions. For a summary of our discussion, refer to Table 2. In addition, we discuss further assertions in the full version of this article.

(Non)-Bipartiteness and k-colorability. We can use a matrix representation to visually prove bipartiteness; see Figure 5a. When sorting the rows and columns according to the two independent subsets, bipartiteness can be simply checked by verifying if the two empty squares are indeed empty [58]. This approach also generalizes to k-colorability as shown in Figure 5b for 4 colors, however, for sparse graphs like in Figure 5c additional highlighting of the (supposedly) empty squares might be necessary. For small graphs, a node-link diagram might be easier to read and hence preferable, however the approach does not scale well due to resolution since the judge needs to verify there are no edges within the subsets; see Figure 5d.

Table 2 Visual proofs in this paper, computational complexity of the problem and perceptual complexity of presented visual proofs (n and m denote the numbers of vertices and edges, resp.).

Assertion	Comp. Complexity	Percep. Complexity	Sec.
G is connected	$O(n+m)$	$O(n)$	2.2
G is *not* (2-)connected		$O(1)$	2.1
G is *not* k-connected	$O(k^3n^2)$	$O(k)$	2.1
G is (not) complete	$O(n^2)$	$O(1)$	5
G has a Hamilt. cycle (path)	NP-complete	$O(1)$	2.3
G has a length-k cycle (path)	NP-complete	$O(k)$	2.3
G is (not) bipartite	$O(n+m)$	$O(1)$	5
G is k-colorable	NP-complete	$O(k)$	5
CoNP-complete assertions	coNP-complete	*Conj.:* No Visual Proof	3.3

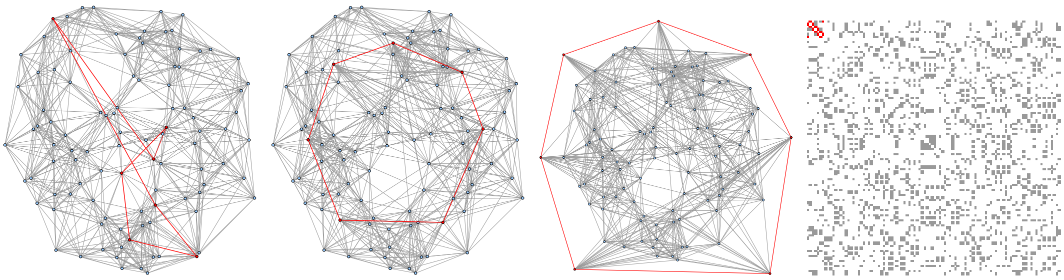


(a) **(b)** **(c)** **(d)**

Figure 5 Visualizing k-colorability. (a) A bipartite graph. (b) A dense 4-colorable graph. (c) and (d) a adjacency matrix and a node-link visualization of a sparse 4-colorable graph.

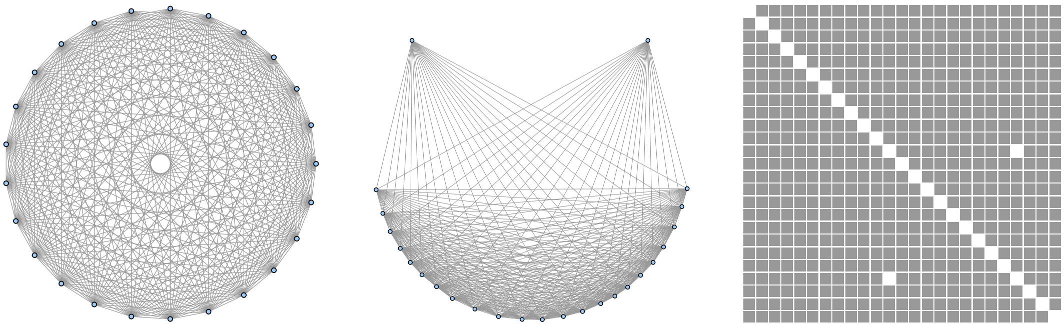
An odd-length cycle certifies that a graph is not bipartite, so non-bipartiteness can be visually proven by highlighting a shortest odd cycle in a drawing. In an arbitrary drawing, the cycle may be hard to spot, see Figure 6a. Redrawing the cycle in convex position makes it easier to read (see Figure 6b), especially if it is the convex outer cycle; see Figure 6c (this makes the rest of the graph harder to read; see Table 1). The cycle is now clearly visible and the judge just needs to assert oddness. While depending on the odd cycle length counting may be inevitable, the judge can use the symmetry of the drawing of the cycle to see that the cycle is odd (e.g., in Figure 6c, there is a single top-most but no single bottom-most vertex). For larger cycle lengths, an adjacency matrix representation may be beneficial: Sort the rows and columns along the odd cycle and mark it, then append the remaining vertices arbitarily. Then, alter the spacing of the matrix so that even rows and columns are thicker than odd ones; see Figure 6d. The cell closing the cycle is a square if and only if the length is odd.

Completeness and Non-Completeness. Non-completeness is evidenced by a single missing edge and can be visually proven with a circular layout with the missing edge on the outer cycle. This approach does not scale well for a larger graphs; see Figure 7a. Readability and scalability can be improved by drawing focus to the missing edge, see Figure 7b. However, one can also use a matrix representation (see Figure 7c) since spotting a missing square scales well from a perception perspective [58]. This technique can also prove completeness.



(a) **(b)** **(c)** **(d)**

Figure 6 Visualizing non-bipartiteness of a graph. In (a) the odd cycle is self-intersecting, making it difficult to certify that it is in fact a cycle. Both in (b) and (c) the cycle is clearly visibile where in (c) the cycle forms the outer boundary of the drawing letting it stand out even more compared to (b). Finally, in (d) the odd cycle is represented by a distinguishable pattern in the adjacency matrix.



(a) **(b)** **(c)**

Figure 7 Visualizing non-completeness of a graph. In (a) the missing edge is very difficult to spot. In (b) and (c) on the other hand it is obvious that an edge is missing.

6 Limitations of the GraphTrials Model

Scalability. In the GraphTrials model, we must not only visualize the evidence represented in the visual certificate, but also display the remainder of the graph faithfully. This may result in higher computational complexity compared to other visualization techniques, e. g., force-directed graph layouts, whose purpose is to create an overall readable representation. Why not forgo visualization completely and use an assertion software to validate the evidence computationally? While this could drastically reduce the computation time and require fewer software components, there are in fact real-world application scenarios, e. g., in court, where it may be better to show a visual certificate accompanied by a short explanation why the certificate is indeed establishing the assertion instead of simply telling the audience that a piece of software analyzed the network and found the evidence for the assertion; see Section 1. Another benefit of visual proofs over a non-visual assertion software is that bugs in the visual proof pipeline can be spotted in the visual certificate, i. e., either the represented graph is not the input graph or the evidence is not a true evidence for the claim.

Another scalability issue is to display the entire graph faithfully. In Section 4, we assumed that the visual certificate may be represented by few components in the judge's mental model and that the formation of that mental model can be mainly guided by usage of bottom-up and pattern recognition processes. For large input graphs, the screen resolution might not permit an information-faithful representation of the input graph so that one must resort to techniques for displaying larger data, e. g., zooming. The introduction of such modes of user interaction may be problematic for our model as it may lead the judge to increasingly use top-down processes of perception which may influence the formation of the mental model.

Human factors. In our model, the judge is necessarily a human actor in the visual proof process. Hence, it is no surprise that human factors play an important role in the application of our model. Our model assumes that the judge is able to draw objective conclusions provided the evidence by the prosecution lawyer. This process may be hindered by insufficient background knowledge of the judge or subjective expectations towards the visualization. Moreover, the judge's mental model cannot be directly analyzed and influenced introducing uncertainty into the model. We discuss these aspects further in the full version of this article.

7 Open Problems

(i) Are visual proofs in fact scalable? How do they extend to geospatial and dynamic graphs where the data are expected to obey spatial and/or temporal constraints? *(ii)* Which features contribute to perceptual complexity? *(iii)* Do response times depend mostly on perceptual complexity? *(iv)* When do human users regard a visual certificate as unimpeachable? *(v)* What are human limits for the perception of graph properties? For instance, the minimum perceivable slope difference is ≈ 2 degrees [31]. *(vi)* What is the trade-off between perceptual complexity and cognitive load?

References

1 Abu Reyan Ahmed, Felice De Luca, Sabin Devkota, Stephen G. Kobourov, and Mingwei Li. Multicriteria scalable graph drawing via stochastic gradient descent, $(\mathrm{SGD})^2$. *IEEE Trans. Vis. Comp. Graph.*, 28(6):2388–2399, 2022. doi:10.1109/TVCG.2022.3155564.

2 Martin Aigner and Günter M. Ziegler. Five-coloring plane graphs. In *Proofs from THE BOOK (3. ed.)*. Springer, 2004. doi:10.1007/978-3-662-05412-3_30.

3 Lorenzo Angori, Walter Didimo, Fabrizio Montecchiani, Daniele Pagliuca, and Alessandra Tappini. Chordlink: A new hybrid visualization model. In Daniel Archambault and Csaba D. Tóth, editors, *Graph Drawing and Network Visualization - 27th International Symposium, GD 2019, Prague, Czech Republic, September 17-20, 2019, Proceedings*, volume 11904 of *Lecture Notes in Computer Science*, pages 276–290. Springer, 2019. doi:10.1007/978-3-030-35802-0_22.

4 Lorenzo Angori, Walter Didimo, Fabrizio Montecchiani, Daniele Pagliuca, and Alessandra Tappini. Hybrid graph visualizations with chordlink: Algorithms, experiments, and applications. *IEEE Trans. Vis. Comp. Graph.*, 28(2):1288–1300, 2022. doi:10.1109/TVCG.2020.3016055.

5 K. Appel and W. Haken. Special announcement. *Discret. Math.*, 16(2):179–180, 1976. doi:10.1016/0012-365X(76)90147-3.

6 Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009. doi:10.1017/CBO9780511804090.

7 Michael Behrisch, Benjamin Bach, Nathalie Henry Riche, Tobias Schreck, and Jean-Daniel Fekete. Matrix reordering methods for table and network visualization. *Comput. Graph. Forum*, 35(3):693–716, 2016. doi:10.1111/cgf.12935.

8 Michael A. Bekos, Henry Förster, Christian Geckeler, Lukas Holländer, Michael Kaufmann, Amadäus M. Spallek, and Jan Splett. A heuristic approach towards drawings of graphs with high crossing resolution. In Therese Biedl and Andreas Kerren, editors, *Graph Drawing and Network Visualization - 26th International Symposium, GD 2018, Barcelona, Spain, September 26-28, 2018, Proceedings*, volume 11282 of *Lecture Notes in Computer Science*, pages 271–285. Springer, 2018. doi:10.1007/978-3-030-04414-5_19.

9 Michael A. Bekos, Henry Förster, Christian Geckeler, Lukas Holländer, Michael Kaufmann, Amadäus M. Spallek, and Jan Splett. A heuristic approach towards drawings of graphs with high crossing resolution. *Comput. J.*, 64(1):7–26, 2021. doi:10.1093/comjnl/bxz133.

10 Irwan Bello, Hieu Pham, Quoc V. Le, Mohammad Norouzi, and Samy Bengio. Neural combinatorial optimization with reinforcement learning. In *Intl. Conf. Learning Represent. ICLR*. OpenReview.net, 2017. URL: https://openreview.net/forum?id=Bk9mxlSFx.

11 Liliana Bounegru, Tommaso Venturini, Jonathan Gray, and Mathieu Jacomy. Narrating networks. *Digital Journalism*, 5(6):699–730, 2017. doi:10.1080/21670811.2016.1186497.

12 James Robert Brown. *Philosophy of Mathematics: A Contemporary Introduction to the World of Proofs and Pictures.* Routledge, 2nd edition, 2008. doi:10.4324/9780203932964.

13 Michael Burch, Kiet Bennema ten Brinke, Adrien Castella, Ghassen Karray, Sebastiaan Peters, Vasil Shteriyanov, and Rinse Vlasvinkel. Guiding graph exploration by combining layouts and reorderings. In Michael Burch, Michel A. Westenberg, Quang Vinh Nguyen, and Ying Zhao, editors, *VINCI: Intl. Symp. Vis. Inform. Comm. Interact.*, pages 25:1–25:5. ACM, 2020. doi:10.1145/3430036.3430064.

14 Stuart K. Card, Jock D. Mackinlay, and Ben Shneiderman. *Readings in information visualization: using vision to think.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1999.

15 Davide Ceneda, Theresia Gschwandtner, and Silvia Miksch. A review of guidance approaches in visual data analysis: A multifocal perspective. *Comput. Graph. Forum*, 38(3):861–879, 2019. doi:10.1111/CGF.13730.

16 Jaegul Choo and Shixia Liu. Visual analytics for explainable deep learning. *IEEE Comput. Graph. Appl.*, 38(4):84–92, 2018. doi:10.1109/MCG.2018.042731661.

17 Sabin Devkota, Abu Reyan Ahmed, Felice De Luca, Katherine E. Isaacs, and Stephen G. Kobourov. Stress-plus-x (SPX) graph layout. In Daniel Archambault and Csaba D. Tóth, editors, *Intl. Symp. Graph Drawing*, volume 11904 of *LNCS*, pages 291–304. Springer, 2019. doi:10.1007/978-3-030-35802-0_23.

18 Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs.* Prentice-Hall, 1999.

19 Emilio Di Giacomo, Walter Didimo, Giuseppe Liotta, and Henk Meijer. Area, curve complexity, and crossing resolution of non-planar graph drawings. In David Eppstein and Emden R. Gansner, editors, *Graph Drawing, 17th International Symposium, GD 2009, Chicago, IL, USA, September 22-25, 2009. Revised Papers*, volume 5849 of *Lecture Notes in Computer Science*, pages 15–20. Springer, 2009. doi:10.1007/978-3-642-11805-0_4.

20 Emilio Di Giacomo, Walter Didimo, Giuseppe Liotta, and Henk Meijer. Area, curve complexity, and crossing resolution of non-planar graph drawings. *Theory Comput. Syst.*, 49(3):565–575, 2011. doi:10.1007/s00224-010-9275-6.

21 Walter Didimo, Luca Giamminonni, Giuseppe Liotta, Fabrizio Montecchiani, and Daniele Pagliuca. A visual analytics system to support tax evasion discovery. *Decis. Support Syst.*, 110:71–83, 2018. doi:10.1016/j.dss.2018.03.008.

22 Walter Didimo, Luca Grilli, Giuseppe Liotta, Lorenzo Menconi, Fabrizio Montecchiani, and Daniele Pagliuca. Combining network visualization and data mining for tax risk assessment. *IEEE Access*, 8:16073–16086, 2020. doi:10.1109/ACCESS.2020.2967974.

23 Walter Didimo, Luca Grilli, Giuseppe Liotta, Fabrizio Montecchiani, and Daniele Pagliuca. Visual querying and analysis of temporal fiscal networks. *Inf. Sci.*, 505:406–421, 2019. doi:10.1016/j.ins.2019.07.097.

24 M. R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, 1979.

25 Robert Gove. Gragnostics: Fast, interpretable features for comparing graphs. In Ebad Banissi, Anna Ursyn, Mark W. McK. Bannatyne, Nuno Datia, Rita Francese, Muhammad Sarfraz, Theodor G. Wyeld, Fatma Bouali, Gilles Venturini, Hanane Azzag, Mustapha Lebbah, Marjan Trutschl, Urska Cvek, Heimo Müller, Minoru Nakayama, Sebastian Kernbach, Loredana Caruccio, Michele Risi, Ugo Erra, Autilia Vitiello, and Veronica Rossano, editors, *Intl. Conf. Inform. Visual. IV*, pages 201–209. IEEE, 2019. doi:10.1109/IV.2019.00042.

26 Robert Gove. Gragnostics: Evaluating fast, interpretable structural graph features for classification and visual analytics. In Boris Kovalerchuk, Kawa Nazemi, Răzvan Andonie, Nuno Datia, and Ebad Banissi, editors, *Integrating Artificial Intelligence and Visualization for Visual Knowledge Discovery*, pages 311–336. Springer, 2022. doi:10.1007/978-3-030-93119-3_12.

27 Kieran Healy. *Data Visualization: A Practical Introduction.* Princeton University Press, 2018.

28 Nathalie Henry and Jean-Daniel Fekete. MatrixExplorer: a dual-representation system to explore social networks. *IEEE Trans. Vis. Comp. Graph.*, 12(5):677–684, 2006. doi:10.1109/TVCG.2006.160.

29 Nathalie Henry, Jean-Daniel Fekete, and Michael J. McGuffin. Nodetrix: a hybrid visualization of social networks. *IEEE Trans. Vis. Comp. Graph.*, 13(6):1302–1309, 2007. doi:10.1109/TVCG.2007.70582.

30 R. Houtkamp, H. Spekreijse, and P. R. Roelfsema. A gradual spread of attention. *Perception & Psychophysics*, 65(7):1136–1144, 2003. doi:10.3758/BF03194840.

31 Maurice Janssen, Marc Lauvenberg, Wesley van der Ven, Twan Bloebaum, and Herman Kingma. Perception threshold for tilt. *Otol Neurotol.*, 32(5):818–825, 2011. doi:10.1097/MAO.0b013e31821c6c7b.

32 Tomihisa Kamada and Satoru Kawai. An algorithm for drawing general undirected graphs. *Inf. Process. Lett.*, 31(1):7–15, 1989. doi:10.1016/0020-0190(89)90102-6.

33 Sylvain Lazard, William J. Lenhart, and Giuseppe Liotta. On the edge-length ratio of outerplanar graphs. In Fabrizio Frati and Kwan-Liu Ma, editors, *Graph Drawing and Network Visualization - 25th International Symposium, GD 2017, Boston, MA, USA, September 25-27, 2017, Revised Selected Papers*, volume 10692 of *Lecture Notes in Computer Science*, pages 17–23. Springer, 2017. doi:10.1007/978-3-319-73915-1_2.

34 Sylvain Lazard, William J. Lenhart, and Giuseppe Liotta. On the edge-length ratio of outerplanar graphs. *Theor. Comput. Sci.*, 770:88–94, 2019. doi:10.1016/j.tcs.2018.10.002.

35 Steven J. Luck and Andrew Richard Hollingworth. *Visual Memory.* Oxford University Press US, 2008. doi:10.1093/acprof:oso/9780195305487.001.0001.

36 Ross M. McConnell, Kurt Mehlhorn, Stefan Näher, and Pascal Schweitzer. Certifying algorithms. *Comput. Sci. Review*, 5(2):119–161, 2011. doi:10.1016/j.cosrev.2010.09.009.

37 R.B. Nelsen. *Proofs Without Words: Exercises in Visual Thinking.* Classroom resource materials. Mathematical Association of America, 1993. URL: https://books.google.de/books?id=Kx2cjyzTIYkC.

38 Quan Hoang Nguyen, Peter Eades, and Seok-Hee Hong. On the faithfulness of graph visualizations. In Sheelagh Carpendale, Wei Chen, and Seok-Hee Hong, editors, *IEEE PacificVis*, pages 209–216, 2013. doi:10.1109/PACIFICVIS.2013.6596147.

39 Takao Nishizeki and Md. Saidur Rahman. *Planar Graph Drawing*, volume 12 of *Lect. Notes Ser. Computing.* World Scientific, 2004. doi:10.1142/5648.

40 Oxford English Dictionary. four-colour | four-color, adj. In *Oxford English Dictionary.* Oxford University Press, 2023. doi:10.1093/OED/3549282820.

41 Peter Pirolli and Stuart Card. The sensemaking process and leverage points for analyst technology as identified through cognitive task analysis. In *Proc. Intl. Conf. Intelligence Analysis*, volume 5, pages 2–4. McLean, VA, USA, 2005.

42 H Purchase. Metrics for graph drawing aesthetics. *J. Vis. Lang. & Comput.*, 13(5):501–516, 2002. doi:10.1016/S1045-926X(02)90232-6.

43 D. Sacha, A. Stoffel, F. Stoffel, B. C. Kwon, G. Ellis, and D. A. Keim. Knowledge generation model for visual analytics. *IEEE Trans. Vis. Comp. Graph.*, 20(12):1604–1613, 2014. 10.1109/TVCG.2014.2346481. doi:10.1109/TVCG.2014.2346481.

44 J. M. Six and I. G. Tollis. A framework for circular drawings of networks. In Jan Kratochvíl, editor, *Proc. 7th Intl. Symp. Graph Drawing*, volume 1731 of *LNCS*, pages 107–116. Springer, 1999. doi:10.1007/3-540-46648-7_11.

45 Sicheng Song, Chenhui Li, Dong Li, Juntong Chen, and Changbo Wang. Graphdecoder: Recovering diverse network graphs from visualization images via attention-aware learning. *IEEE Trans. Vis. Comp. Graph.*, pages 1–17, 2022. doi:10.1109/TVCG.2022.3225554.

46 Roberto Tamassia, editor. *Handbook on Graph Drawing and Visualization.* Chapman and Hall, 2013. URL: https://www.crcpress.com/Handbook-of-Graph-Drawing-and-Visualization/Tamassia/9781584884125.

47 Edward Rolf Tufte. *The Visual Display of Quantitative Information.* Graphics Press, 1992.

48 Edward Rolf Tufte. *Visual explanations - images and quantities, evidence and narrative.* Graphics Press, 1997. URL: https://www.worldcat.org/oclc/36234417.

49 Edward Rolf Tufte. *The Visual Display of Quantitative Information.* Graphics Press, 2001.

50 Thomas Tymoczko. The four-color problem and its philosophical significance. *J. Philosophy*, 76(2):57–83, 1979.

51 Bret Victor. Scientific communication as sequential art. http://worrydream.com/ScientificCommunicationAsSequentialArt/, 2011.

52 Johan Wagemans, James H. Elder, Michael Kubovy, Stephen E. Palmer, Mary A. Peterson, Manish Singh, and Rüdiger von der Heydt. A century of gestalt psychology in visual perception: I. perceptual grouping and figure–ground organization. *Psychol. Bulletin*, 138(6):1172–1217, 2012. doi:10.1037/a0029333.

53 Colin Ware. *Information Visualization – Perception for Design.* Morgan Kaufmann, 2004.

54 Colin Ware. *Visual Thinking For Information Design.* Morgan Kaufmann, 2021. doi:10.1016/C2016-0-01395-5.

55 Duncan J. Watts and Steven H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, 1998. doi:10.1038/30918.

56 Hadley Wickham, Dianne Cook, Heike Hofmann, and Andreas Buja. Graphical inference for InfoVis. *IEEE Trans. Vis. Comp. Graph.*, 16(6):973–979, 2010. doi:10.1109/TVCG.2010.161.

57 Wikipedia. Mathematical proof. https://en.wikipedia.org/wiki/Mathematical_proof#Visual_proof, 2023.

58 Jeremy Wolfe. Visual search. *Cur. Biol.*, 20(8):R346–R349, 2010. doi:10.1016/j.cub.2010.02.016.

59 yWorks. yEd – graph editor. https://www.yworks.com/products/yed, 2023.