Check for updates

E^2Storyline: Visualizing the Relationship with Triplet Entities and Event Discovery

YUNCHAO WANG, GUODAO SUN, ZIHAO ZHU, TONG LI, LING CHEN, and RONGHUA LIANG, Zhejiang University of Technology, China

The narrative progression of events, evolving into a cohesive story, relies on the entity-entity relationships. Among the plethora of visualization techniques, storyline visualization has gained significant recognition for its effectiveness in offering an overview of story trends, revealing entity relationships, and facilitating visual communication. However, existing methods for storyline visualization often fall short in accurately depicting the specific relationships between entities. In this study, we present E^2Storyline, a novel approach that emphasizes simplicity and aesthetics of layout while effectively conveying entity-entity relationships to users. To achieve this, we begin by extracting entity-entity relationships from textual data and representing them as subject-predicate-object (SPO) triplets, thereby obtaining structured data. By considering three types of design requirements, we establish new optimization objectives and model the layout problem using multi-objective optimization (MOO) techniques. The aforementioned SPO triplets, together with time and event information, are incorporated into the optimization model to ensure a straightforward and easily comprehensible storyline layout. Through a qualitative user study, we determine that a pixel-based view is the most suitable method for displaying the relationships between entities. Finally, we apply E^2Storyline to real-world data, including movie synopses and live text commentaries. Through comprehensive case studies, we demonstrate that E^2Storyline enables users to better extract information from stories and comprehend the relationships between entities.

CCS Concepts: • **Human-centered computing → Visualization theory, concepts and paradigms**; *HCI theory, concepts and models*; *User studies*; *Information visualization;*

Additional Key Words and Phrases: Layout, information visualization, multi-objective optimization, storytelling visualization

ACM Reference format:
Yunchao Wang, Guodao Sun, Zihao Zhu, Tong Li, Ling Chen, and Ronghua Liang. 2024. E^2Storyline: Visualizing the Relationship with Triplet Entities and Event Discovery. *ACM Trans. Intell. Syst. Technol.* 15, 1, Article 16 (January 2024), 26 pages.
https://doi.org/10.1145/3633519

The work is partly supported by the National Key Research and Development Program of China (grant no. 2020YFB1707700), the Zhejiang Provincial Natural Science Foundation of China (grant nos. LR23F020003 and LTGG23F020005), the National Natural Science Foundation of China (grant nos. 61972356 and 62036009), and the Fundamental Research Funds for the Provincial Universities of Zhejiang (grant no. RF-B2023006).
Authors' address: Y. Wang, G. Sun, Z. Zhu, T. Li, L. Chen, and R. Liang, Zhejiang University of Technology, No. 288, Liuhe Road, Hangzhou, China; e-mails: wyctears@gmail.com, guodao@zjut.edu.cn, iszhuzh@outlook.com, litong@zjut.edu.cn, 2899719394@qq.com, rhliang@zjut.edu.cn.

2157-6904/2024/01-ART16 $15.00
https://doi.org/10.1145/3633519

1 INTRODUCTION

Among text data such as novels and movie synopses, there are stories that consist of one or more events. A *story* is a general term for a series of *events* that are related in time or cause and effect [1, 2]. Effectively showing the plots of stories and the relationships between *characters* has been a hot research topic in the field of visualization [3]. Some existing visualization techniques perform remarkably well in presenting specific analytical tasks, e.g., the evolution of plots or topics (flow diagrams) [4, 5], connections between events or characters (node-link diagrams) [6, 7], and spatio-temporal information about where stories occur (scatter diagrams) [8, 9]. Nonetheless, these visualization techniques fail to display comprehensive information. Flow diagrams give users overviews of evolution of plots or topics in stories, but the display of characters is insufficient (detail); node-link diagrams make it difficult to take into account spatio-temporal information (clutter); and scatter diagrams tend to lose a grasp of the whole (overview). In *Munro*'s [10] hand-drawn film narrative drawings, a visual technique for illustrating the evolution of stories, *storyline visualization*, emerged. There are several visual designs in his hand drawings: from left to right representing the progression of the stories and lines to encode the entities. Storyline visualization, with its simplicity, intuitiveness, and effectiveness, is often used to show story trends (wiggle) and relationships between characters (cluster or diverge) [11–13].

While users can understand some behavioral patterns of entities throughout the story through the aggregation and dispersion of lines, it is limited to "*whether the entities are involved in the same scene or location*" [14, 15]. It means that users fail to know the specific behavior between entities in a given situation, such as during an event or at a moment. In other words, the above work focuses on the presentation of the overall trend of stories. To this end, we believe that showing general trends of stories should be balanced with showing the *entity–entity relationships*. Here, we define "the interaction between two entities" as "the entity–entity relationship". Taking the sentence "Tom ate two oranges at home this morning" as an example, there are two entities, "Tom" and "oranges", and their interaction is "ate". In simple terms, the entity–entity relationship in this sentence is "Tom"-"ate"-"oranges". In this article, we use **natural language processing (NLP)** tools to extract triplets with the structure <***Subject-Predicate-Object***> **(SPO)** from text data, and use them to approximate the entity–entity relationships. By adding the SPO triplets, users can analyze the behavioral patterns of individual entities while also understanding their relationships with others.

The data structure displayed in the storyline visualization is a temporal graph. Thus, in a sense, storyline visualization is a subset of dynamic graph visualization techniques. An event that occurs at a certain moment in a story can be viewed as the graphical structure of a dynamic graph at that moment. The entities involved in the event, as well as the entity–entity relationships, are represented as nodes and edges in the graph structure. There are two typical representatives of dynamic graph visualization techniques: node-link diagram and adjacency matrix. The former displays the entity–entity relationships through links. Some works attempt to incorporate temporal aspects into the node-link diagram and, despite optimizing the layout, it still generates complex topological structures in the overall layout, making it difficult to understand and read [16]. The latter encodes entities at a certain stage into rows and columns to generate a static adjacency matrix. The static adjacency matrix can visually represent the existence of entity–entity relationships (shaded or marked). Through supplemental visualizations or effects (e.g., animations), static adjacency matrices can also convey evolutionary trends to the user [17–19]. However, both static node-link diagrams and static adjacency matrices fail to provide detailed representations of the entity–entity relationships at a specific stage, particularly the sequence of relationships occurring among entities within that stage. We propose a visual design for presenting the entity–entity relationships that is suitable for storyline visualization. With reference to the static adjacency matrix, we encode time as columns of the matrix and entities as rows. If two cells in a column of the matrix are colored, it means that there is a relationship between the entities corresponding to these two cells. Experiments show that this visual design fits with the generic form of storyline visualization. In order to make the layout concise, we propose an effective **multi-objective optimization (MOO)** algorithm. The algorithm takes into account different optimization objectives, both global and local. For the global, we need to minimize line crossings and oscillations as much as possible. For the local, we need to make the related entities as close to each other as possible. The modeling is performed by determining the objective function and constraints to optimize the layout of the storyline visualization.

Specifically, the contributions of our work are as follows.

— A novel form of storyline visualization that shows more details. Introducing events and entity–entity relationships increases the quantity and quality of information presented.
— A multi-objective optimization model that combines multiple design requirements to create a clean and intuitive layout for storyline visualization.
— Several case evaluations based on movie synopses and live text commentaries demonstrate the usability and effectiveness of E^2Storyline.

2 RELATED WORK

2.1 Storyline Visualization

Storyline visualization is one of the common visualization techniques used in the visualization of temporal sequences. Storyline visualization was originally proposed in *movie narrative charts* [10] to show the connections between characters and story development in movies. Given the effectiveness and intuitiveness of storyline visualization in movie overviews, this visualization technique has been applied to other data and domains, such as software evolution [20], meeting minutes [21], and group collaboration [22]. Furthermore, for the design space of storyline visualization, Tang et al. [13] proposed a set of design considerations to make it more aesthetically pleasing and easy to read. By collecting lots of hand-drawn works, *iStoryline* researched and summarized how users could convert narratives into hand-drawn storylines. It proposed a design space and developed an authoring tool to achieve a balance between hand-drawn storyline and automatic layout. Taking it a step further, based on a reinforcement learning framework to train artificial intelligence, *PlotThread* [23] assisted users in exploring the design space and generating optimized storylines.

Building upon the significance of aesthetic and informative qualities, metaphors have become a crucial aspect in visualization. By blending validity and aesthetics, metaphors offer a means to convey extensive information while maintaining an appealing visual presentation. An example of this can be observed in the use of color-filled background areas to represent hierarchical information, as demonstrated in the case of *SplitStreams* [24], which introduces a novel visual metaphor for providing a static overview of temporal hierarchy. Visual metaphors can also extend beyond conventional techniques and take the form of prominent artworks [25].

Researchers in the field of visualization have been pursuing simplicity and readability of diagrams. Therefore, layout optimization is also frequently mentioned in storyline visualization. A common idea is to improve the position, size, and shape of the elements in the diagram according to certain principles, i.e., to optimize the layout of the elements in the diagram. The optimization objectives involved in many current layout optimization efforts are common across multiple visualization diagrams, namely: (1) reducing line wiggles, (2) reducing line crossings, and (3) reducing blank areas [11, 12, 26]. The optimization objectives of the metro map are very similar to those of the storyline visualization: for example, modeling the line crossing minimization problem as a multi-layer minimization problem with tree constraints [27] or using mixed integer programming to optimize the metro map layout and marker locations to reduce blank areas [28]. The application of metro map visualization to set visualization in *MetroSet* [29] facilitated the understanding of sets. Although the optimization objectives are similar, their origin, i.e., the design requirements, are distinct. In storyline visualization, the design requirements are: (1) lines of the same set (with relationships) should be close to each other and (2) lines need to be maintained until the set (relationship) in which they are located changes. In Section 3.2, we will expand on the design requirements in detail.

2.2 Visualization of Temporal Events

Temporal event visualization is a time-centric visualization method employed for depicting the temporal evolution of events. We have synthesized renowned contributions in the field of temporal event visualization, encompassing guidelines, overviews, and detailed analyses. The section on guidelines primarily encompasses theoretical and empirical aspects, whereas the overview section offers a holistic visual portrayal of the entire temporal event and the details section emphasizes the visualization of individual event attributes.

For advanced visual analysis methods applied to temporal event data, researchers have proposed a comprehensive survey to describe and classify them, referred to as the design space [30]. In the context of storytelling in visualization, Tong et al. [31] conducted research on this topic, providing an overview of common and crucial elements as well as presenting various challenges in the field. Of particular interest in temporal event analysis is the investigation of narrative order. Kim et al. [32] introduced a pre-order classification of nonlinear narratives and developed a visual analysis system utilizing stacked diagrams. Subsequently, Lan et al. [33] contributed a detailed classification of narrative order in stories, leveraging crowdsourcing to explore users' perceptions of different narrative orders. These aforementioned guidelines not only offer a design space and visualization elements for organizing temporal event visualization but also initiate a theoretical discourse on the discrepancy between narrative sequence and event occurrence time in this context. However, it should be noted that the positioning of events on the timeline does not necessarily correspond to the actual timing of their occurrence. To be more precise, events in storyline visualization are arranged according to the order in which they are mentioned in the story. Although the narrative order may not align with chronological order, it still captures the plot development of the story.

Timelines used for visualizing temporal events are presented in diverse formats, including straight lines, spirals, and concentric circles. Di Bartolomeo et al. [34] have provided design guidelines for developing effective timeline visualizations based on user tasks and data types, considering different forms of timeline representation. They have verified that the shape of the timeline (e.g., line, circle, spiral) has an impact on task performance. Through user feedback, it was found that the preferred form for representing timelines is a straight line. For instance, *LifeLine* [35] employed a straight line representation where events are positioned on the timeline based on their occurrence. In the case of *StoryCake* [2] and *StoryPrint* [36], scenes and characters in movies are visually represented using concentric rings in circular timelines to enable comparative analysis. In addition to these commonly used approaches, *Time Curves* [37] offered an effective method for researchers to gain an overview of temporal data evolution. A graph-based overview can also assist researchers in capturing crucial points during temporal event analysis and facilitating a better understanding of the development [6, 7]. In this study, there is no explicitly defined axis to represent time. Instead, the progression of the story development is depicted by plotting lines from left to right.

Methods applied to analyze complex event sequences include aggregation and detailed display, among others. Common forms of aggregation include Sankey diagrams [38, 39], icicle diagrams

[40, 41], and transfer matrices [42–44]. The aforementioned visualization forms follow the same display order, from the beginning to the end of the story. Although they can visualize the arrangement of all events, they have poor scalability and cannot display detailed levels. By constructing a story overview [45, 46], selecting events or time [41, 47, 48], or defining some rules [49, 50], the scale of events can be reduced. Users can then explore and analyze details with the help of interactive methods [51, 52]. In order to bridge the gap between visual analysis and storytelling, Chen et al. [53] proposed an overall framework that combines data analysis and result presentation through story synthesis. In storyline visualization, nodes are frequently employed to display more detailed information. Nodes can be segmented by stages [32, 47, 54] or periodic time divisions [55–58]. Guo et al. [47] used an unsupervised algorithm to align and segment event sequences by stages. They further aggregated the segments within each stage into clusters and illustrated the evolutionary patterns between clusters at different stages. *TimeArcs* [16] segmented the data on a yearly basis. If there was collaboration between authors within a year, they were connected using Bessel curves. *Sequence Braiding* [59] determined the nodes based on the periodicity of eating. They used a layered directed acyclic network to visualize the overall event time sequence and attributes. At each node, sequence braiding sorted them according to predefined rules to facilitate the observation of patients' blood glucose changes. Inspired by the aforementioned work, we describe several attempts made to illustrate the entity–entity relationships in Section 5.

The above works do not simultaneously show an overview of story trends and details of the entity–entity relationships. Therefore, we adopt hybrid visualization to solve this problem. Hybrid visualization can combine multiple visualization techniques to display various data views simultaneously in a single visual interface, helping people gain a better understanding and analysis of the data. Hybrid visualization is applied in community analysis [17], narrative spatio-temporal data visualization analysis [60], large dataset clustering analysis [61], and event sequence branch pattern analysis [62]. *NodeTrix* [17] was a hybrid representation for network that combines the advantages of node-link diagrams and static adjacency matrices. To extract and visualize branching patterns from event sequences, *CoreFlowVis* [62] had a hybrid design combining features of node-link visualization and icicle plots. Due to the effectiveness of hybrid visualization in conveying information, this article uses the same approach. We combine storyline visualizations with pixel grids to show the evolution of the story while conveying to users the entity–entity relationships in the event. However, how to present this hybrid visualization above in a concise way is a challenge. To solve this issue, we model the layout optimization problem with MOO. By solving the model, the layout that satisfies the optimization objective is obtained. The algorithm is described in Section 4.2.

3 CONSIDERATIONS

Incorporating events into storyline visualization and the relationships between various entities in the event allows for a more intuitive presentation of the information contained in the story, better explanation of the relationships between entities, and the roles of each entity. For this purpose, we propose E^2Storyline, whose goal is to generate storyline visualizations that are legible, intelligible, and demonstrate the actual relationship among entities. In this section, the considerations for input data and visualization design are illustrated.

3.1 Target Data

Narratives can be disseminated through diverse media channels, including books, magazines, films, blogs, forums, emails, and others. This implies that narratives can rely not only on textual content but also on visual imagery and video formats. However, given that the primary objective of this study is to authenticate and assess the applicability of E^2Storyline independent of the data format, text-based data will be utilized and examined within this research. Illustrated in Figure 1, the data

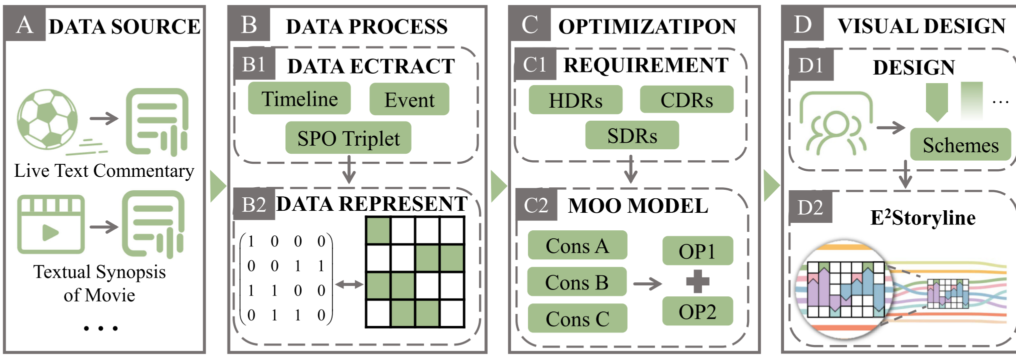


Fig. 1. The pipeline of E^2Storyline. Text data with a strong linear timeline is selected as the basis of this article. The data process stage includes the extraction of SPO triplets, timeline, and event. The optimization stage follows three design requirements, proposes optimization objectives and corresponding constraints, and further realizes the optimal solution. The visual design stage includes a storyline layout based on the entity–entity relationships.

may encompass real-time textual commentaries or textual summaries of movies. The rationales for selecting this particular data type are outlined as follows.

▲ **Linear timeline.** Story development is temporal in nature. The timeline is an essential dimension in the development of stories. To make a story more attractive, the narrative order can follow a chronological order or an anachronistic one in artistic literary works such as novels and movies. However, in emerging media such as social media and live text commentary, time is usually linear. Textual content with good linear time can be better presented with storyline visualization.

▲ **Convenient event extraction.** With the development of artificial intelligence, it has become possible to differentiate and extract events from audio- and video-based data sources, such as movies, TV shows, and radio stations. However, it costs less to get similar results with text-based data.

▲ **Relation encoding.** The relationship between two entities in a text-based story usually needs to be described in long sentences. Such expressions are not concise enough. Therefore, we need to find an efficient way to encode the data. The new approach needs to have the following properties: ❶ **Structured form**. It has a structured form. The structured form reduces the storage space for text and makes the data easier to handle. ❷ **Recorded entity**. It can record the entities involved. The recorded entities can be useful in subsequent model exploration. ❸ **Express relationships**. It can express the relationships between entities. Preserving the entity–entity relationships reduces information loss and makes it easier to discover the behavior patterns of entities. In NLP, a triplet with the structure <*Subject-Predicate-Object*> **(SPO)** [63] can be used to encode sentences with semantic data. An SPO triplet contains both Agent (*Subject*) and Patient (*Object*) of Verb (*Predicate*). Despite the SPO triplets being extracted discretely, they still have temporal information and therefore will not interfere with the overall trend of the story. In view of this, we can perform SPO triplets extraction for entities and actions in the text sentence by sentence.

In Section 6, we evaluate the algorithm with two real datasets that meet the above requirements. One is live text commentary of "*The Champions League* 2015–2016 Round of 16, Barcelona vs. Arsenal first leg". We regard the match highlights that are selected by professionals as events. The other is movie text synopses of "*The Chronicles of Narnia: The Lion, the Witch and the Wardrobe*", "*Rumble in the Bronx*", and "*The Matrix*". We label the events according to the synopses of movies on Wikipedia.

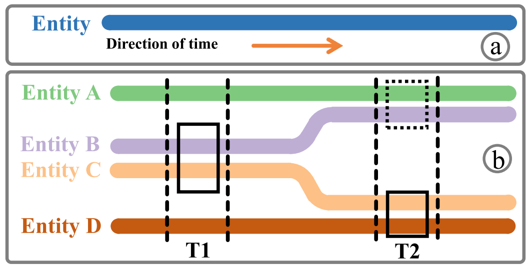


Fig. 2. Schematic diagram of Hard Design Requirements (**HDR**s). (a) Entities are encoded with lines and time flows from left to right. (b) The aggregation and dispersion of entities connotes the evolution of the story.

3.2 Design Requirements

As mentioned in Section 2.2, design requirements are prevalent and critical in various types of visual presentations [11]. By reviewing related papers on storyline visualization, we propose three different levels of design requirements based on the degree of generality of the requirements: Hard design requirements, Soft design requirements, and Custom design requirements.

3.2.1 Hard Design Requirements. Two design requirements for storyline visualization are conventional: "encoding entities with lines" and "aggregation and dispersion of entities". We define them as **Hard Design Requirements** (**HDR**s) which are considered as default design requirements for storyline visualization [10–13, 26].

❖ **HDR1: Connecting the same entities at different moments with curved segments.** Taken as a whole, these curve segments extend from left to right as time progresses.

❖ **HDR2: Using the aggregation and divergence of lines to indicate whether entities participate in the same event.** If certain entities participate in the same event at a certain time, the lines representing those entities will come together. Otherwise, it will diverge, or there will be no obvious aggregation process.

Figure 2(a) illustrates the HDR 1. The blue line represents an entity; the time progression of storyline is plotted from left to right, as shown by the orange arrows. Figure 2(b) illustrates the HDR 2. The 4 entities are encoded with four colors, respectively. In time period T1, Entity B and Entity C are associated; therefore, they are clustered together. In time period T2, the association between Entity B and Entity C no longer exists, whereas the associations between Entity A and Entity B and Entity C and Entity D exist. Therefore, the previous aggregated Entity B and Entity C are separated, and the two latter groups are aggregated with each other.

3.2.2 Soft Design Requirements. In the domain of visualizing storylines, a common objective is to address visual distractions and enhance the conciseness and intuitiveness of the storyline visualization. In the context of storyline visualization, line crossings and line wiggles pose significant visual distractions since entities are represented using lines. Approaches such as *StoryFlow* [12] employed a two-stage method to mitigate these distractions: firstly, a set of relationship trees is generated, and then sorting and line alignment techniques are applied. Similarly, *Sequence Braiding* [59] utilized strategies such as "rank assignment", "graph construction", "within-rank node sorting", and "linear programming for exact intersection reduction" to accomplish the same objective. We define these design requirements (line crossings and line wiggles) that have the same objective but are implemented in different ways as **soft design requirements** (**SDR**s). Generally speaking, the SDRs for storyline visualization are specified as follows:

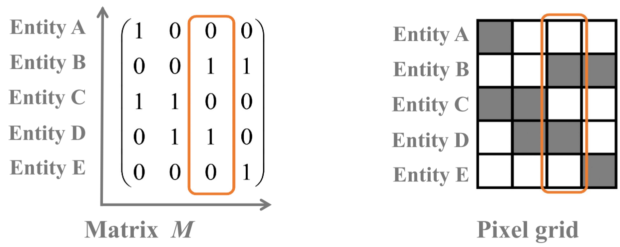


Fig. 3. Schematic diagram of Custom Design Requirements 1 (**CDR**1). The matrix on the left represents an event. The rows of $M_{(x,y)}$ represent different entities, and the columns represent SPO triplets. The pixel grid on the right is a mapping of $M_{(x,y)}$. The orange box identifies the 3rd SPO triplet in event $M_{(x,y)}$.

- **SDR1: Reduce the number of line crossings.** Fewer line crossings ease the reading burden on the user by reducing the complexity of views, thus allowing better understanding of the information contained in them.
- **SDR2: Reduce the number of line wiggles.** Excessive or unnecessary line wiggles not only make lines appear discontinuous visually but also may increase the number of line crossings.

3.2.3 Custom Design Requirements. Due to the diversity of areas where storyline visualization is applied, the focus of datasets and visual representation varies. We define design requirements other than HDRs and SDRs as **custom design requirements** (**CDR**s). *Sequence Braiding* [59] focused on the treatment of type 1 diabetes; therefore, some special requirements were gathered by researchers as the basis of design, implementation, and layout algorithm. To keep the y-axis with a consistent meaning, Arendt et al. [64] proposed some CDRs, i.e., lines corresponding to individuals with the same interaction context and timestep are plotted on the same coordinate points, thus producing a storyline visualization with less misleading. In this paper, we propose two CDRs in order to highlight the relationships between entities in events.

- **CDR1: Convey event-related entities intuitively.** An intuitive representation of entities involved in the same event enhances user understanding and engagement.
- **CDR2: Maintain proximity of SPO triplet entities.** This ensures that entities of the same SPO triplet in the same event are kept close and that there is ease of interpretation.

4 VISUALIZATION TECHNIQUES

In this section, we first do visual mapping of events and SPO triplets according to CDRs presented in Section 3.2.3. Then, we explain SPO triplets extraction technique with the help of NLP tools. Furthermore, we demonstrate the implementation of the layout optimization algorithm in detail.

4.1 Visual Mapping of Events and SPO Triplets

According to CDR1, we use a matrix $M_{(x,y)}$ to metaphorize an event, as shown on the left side of Figure 3. The entities are encoded into rows, and the index of the rows indicates their order in the matrix. The sentences are encoded as columns, and the indexes of the columns are increased sequentially with the order of the sentences. If an SPO triplet can be extracted from a sentence, the position that is determined by the sentence's corresponding column and the entity's corresponding row will be assigned as "1" in the matrix. To further illustrate, we will give a case in point. The orange box in Figure 3 represents the 3rd SPO triplet ($x = 3$) in event $M_{(x,y)}$. In this triplet, Entity A ($i_A = 1$) and Entity C ($i_C = 3$) are related. In summary, we assign values to the corresponding positions in $M_{(x,y)}$, i.e., $M_{(1,1)} = 1$, $M_{(3,1)} = 1$. To visualize $M_{(x,y)}$, we introduce a pixel grid, as

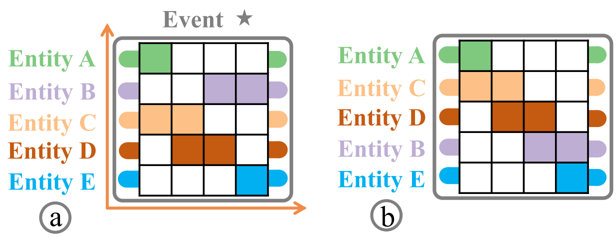


Fig. 4. Schematic diagram of Custom Design Requirements 2 (**CDR**2). (a) The x-axis represents the order of SPO triplets, and the y-axis represents the order of entities. (b) Optimize entities ordering by considering the relationship between them that makes the pixel grid compact.

shown on the right side of Figure 3. The number of rows and columns of the pixel grid is the same as $M_{(x,y)}$. The value of "1" in the matrix is filled with color in the pixel grid, whereas the value of "0" is not. Here, we use different colors to conveniently distinguish entities from each other, as shown in Figure 4.

Keeping events compact makes it easier for users to discover related entities. The CDR2 that we purpose are illustrated in detail in Figure 4. The meaning of x-axis and y-axis in Figure 4(a) is explained in CDR1. The pixel grid in the gray box represents Event ★, and we have integrated both entities and SPO triplets information into it (filled with different colors). In order to achieve CDR2, we need to adjust the ordering of entities (y-axis). Figure 4(b) shows one of the results that satisfies CDR2 after adjusting the row ordering. We can find that the complexity of the pixel grid is significantly reduced so that the readability of events is enhanced. Users can better understand the relationships between entities in an event and the role of a specific entity.

4.2 Entity–Entity Relationship Extraction

As mentioned in Section 3.1, the SPO triplets are very suitable to highlight the entity–entity relationships. Encoding entities and entity–entity relationships with the SPO triplets can structure the data to reduce data volume and speed up computing efficiency. This section will explain the concepts involved and briefly illustrate how we obtain the SPO triplets based on tools of NLP.

In NLP, the triplet extraction is an important fundamental task. The existing NLP tools basically provide the function of triplet extraction, such as *FudanNLP*, *NLTK*, *HanLP*, etc. After comprehensive consideration, *HanLP* [65] is chosen in this article. With the help of *HanLP*, we obtain the SPO triplets by analyzing the dependency relations and semantic dependencies. The former focuses on syntactic structure, whereas the latter focuses on semantic association.

► The first step is raw text cleanup. The formatting of sentences and the use of punctuation are not particularly rigorous in some text data, especially in live text comments. Therefore, characters, punctuation marks, broken spaces, etc., need to be handled.

► The second step is syntactic tree construction. We process the cleaned data with the help of HanLP. The functions in HanLP — dependent syntactic analysis and semantic dependency analysis — can convert sentences into dependency trees. The result of dependency analysis is a directed graph ***G=(V,A)***. ***V*** represents a node, that is, each word in the sentence; ***A*** represents a directed edge, representing the dependency between words. For the example, *"Tom eats oranges."*, its structure is shown in Figure 5. From this, we can see that the dependency syntactic tree (T_{dst}) identifies the syntactic components and the semantic dependency tree (T_{sdt}) identifies the semantic relations. The explanation of tags in Figure 5 is provided in Table 1.

► The third step is the SPO triplets extraction. The dependency syntactic tree and the semantic dependency tree possess distinct characteristics, yet both designate verbs within the sentence as

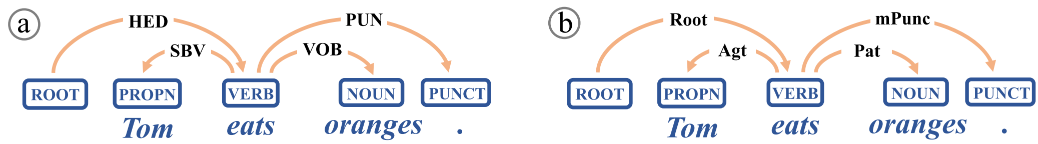


Fig. 5. The dependency tree of the sentence *"Tom eats oranges today"*. The orange arcs refer to the type of relationship between words. In the blue box is the lexical type of the corresponding word. (a) Dependency syntactic tree (T_{dst}). (b) Semantic dependency tree (T_{sdt}).

Table 1. Explanation of Tags in T_{dst} & T_{sdt}

T_{dst}		T_{sdt}	
HED	root of a sentence	Root	root of a sentence
SBV	subject	Agt	agent
VOB	direct object	Pat	patient
PUN	punctuation marker	mPunc	punctuation marker

their respective root nodes (as illustrated in Figure 5, in which the verb "eats" functions as the root node). Integrating these two syntactic trees allows for a more effective extraction of parsing entity-related semantic information [66, 67]. We filter the nodes that satisfy the **subject–verb (*SBV*) and verb–object (*VOB*)** relations from the dependency syntactic tree and the nodes whose semantic relations are "Agent" (*Agt*) and "Patient" (*Pat*) from the semantic dependency tree. By combining both syntactic trees, we could verify the consistency of the nodes and obtain the SPO triplets.

In the example sentence, the result of the dependency syntactic analysis is *SBV*: *Tom*←*eats*, *VOB*: *eats*→*oranges*; the result of the semantic dependency analysis is *Agt*: *Tom*←*eats*, *Pat*: *eats*→*oranges*. It can be seen that the SPO triplet of this sentence is <*Tom-eats-oranges*>. After the text data is processed through the above process, the SPO triplets can be obtained. A detailed illustration of this algorithm is given in Algorithm 1.

4.3 Layout Algorithm Implementation

Following the design requirements proposed in Section 3.2, we discuss the implementation of the layout algorithm from the selection of optimization objectives and mathematical expression of the layout model.

4.3.1 Selection of Optimization Objective. Showing the entity–entity relationships in storyline visualization is a challenging endeavor. Existing efforts on storyline visualization mainly focus on how to provide concise layouts, i.e., layout optimization. However, the optimization algorithms may vary greatly due to different data types and display intently. In other words, algorithms that only satisfy HDRs and SDRs are different from those that cover all three design requirements. The former may focus on a particular requirement (e.g., reducing intersections) [27, 68–70], whereas the latter is considered in a more comprehensive manner [11–13, 26]. Each entity is encoded as a continuous line in HDR1, but the entities are discrete between each other at the same moment. Therefore, determining the physical location at the same time can be formulated as an "*Allocation Problem*". Among the listed design requirements, SDR1, SDR2, and CDR2 are designed to reduce the complexity of view and help the user to observe it better. Although SDR1 and SDR2, respectively, target reducing the complexity of points and lines, they both ultimately reduce the number of crossings. In CDR2, the entities of the same SPO triplet in the same event are as close as possible to each other in order to highlight the two entities that are related—that is, minimizing the sum of the distances between all entities in SPO triplets. We take the reduction of line crossing as

Table 2. Table of Parameters

Symbol	Definition	Notes
H	The length of entity ordering.	
h	A position in entity ordering.	$h \in [1, H]$
T_e	The duration of event e	
$o_{i,t,h}$	Determine whether entity i is at position h at moment t.	0-1 decision variable
$p_{i,j,t}$	Determine whether entity i is above entity j at moment t.	0-1 decision variable
$C_{i,e}$	Determines whether entity i is involved in event e.	0-1 constant
$B_{e,t}$	Determines whether event e is occured in moment t.	0-1 constant

ALGORITHM 1: Extracting SPO Triples

Require: A dependency syntactic tree T_{dst}, a semantic dependency tree T_{sdt}, a two-dimensional array Arr_1 for storing triplets of T_{dst}, a two-dimensional array Arr_2 for storing triplets of T_{sdt}
Ensure: SPO Triplets
  Step 1: Use in-order traversal for T_{dst}.
  if the relationship between a node and its two leaf nodes is SBV and VOB, respectively, **then**
    store the node and its leaf nodes in Arr_1.
  end if
  Step 2: Use in-order traversal for T_{sdt}.
  if the relationship between a node and its two leaf nodes is Agt and Pat, respectively, **then**
    store the node and its leaf nodes in Arr_2.
  end if
  Step 3: Compare Arr_1 with Arr_2.
  if the same nodes exist in Arr_1 and Arr_2. **then**
    combining nodes to form SPO triplets.
  end if

optimization objective 1 (op1) and take the associated entities close to each other as optimization objective 2 (op2). Based on these, we transform the layout optimization problem as the MOO model.

4.3.2 *Mathematical Expression.* After selecting the optimization objectives, we need to transform them into proper mathematical expressions. In the following, we will mathematically explain each of two optimization objectives, *line crossing* and *entire distance*, which are proposed in Section 4.3.1. The symbols and definitions of the parameters involved in the mathematical expression are shown in Table 2.

■ **Line Crossing.** Firstly, we define "*entity ordering*" as the vertical arrangement of entities from top to bottom at a certain moment t. As illustrated in Figure 4(a), entity ordering is listed as [A, B, C, D, E]; in Figure 4(b), entity ordering is listed as [A, C, D, B, E]. We define two entities as i and j. By comparing the y-axis coordinates of entities i and j at adjacent moments, we can determine whether the two lines cross each other. We define $y_{i,t}$, which denotes the y-axis coordinates of entity i at moment t. Similarly, define $y_{j,t}$. If the two lines intersect between moments t and $t+1$, then $(y_{i,t} - y_{j,t})(y_{i,t+1} - y_{j,t+1}) < 0$. To minimize the number of intersections in the whole storyline visualization, op1 can be expressed as

$$min \quad \frac{1}{2} \sum_{i,j} \sum_{t} \mathbb{I}_{\{(y_{i,t}-y_{j,t})(y_{i,t+1}-y_{j,t+1})<0\}}. \tag{1}$$

Considering that Equation (1) is quite complex, we introduce $p_{i,j,t}$ and $o_{i,t,h}$. Both are 0-1 decision variables. The former is used to determine whether the entity i is below the entity j at moment

t. If $y_{i,t} < y_{j,t}$, $p_{i,j,t} = 1$; otherwise, $p_{i,j,t} = 0$. The latter is used to determine the position of entity i in entity sequence at moment t. If the position of entity i is h, $o_{i,t,h} = 1$; otherwise, $o_{i,t,h} = 0$, where $h \in [1, H]$, H means the length of entity sequence. The mathematical relationship between $o_{i,t,h}$ and $y_{i,t}$ is expressed as

$$y_{i,t} = \sum_h o_{i,t,h} \cdot h. \tag{2}$$

Using $p_{i,j,t}$, we can concise Equation (1) as

$$\min_p \quad \sum_{i,j,t} p_{i,j,t} p_{j,i,t+1}. \tag{3}$$

■ **Distance between entities.** Decomposing op2 in time, it can be described as minimizing the difference in y-axis coordinates between entity i and j. Equation (2) illustrates the y-axis coordinates of entity i at moment t. Thus, we can calculate the difference between the y-axis coordinates of the two entities using the following equation.

$$(y_{i,t} - y_{j,t})^2 = \left(\sum_h o_{i,t,h} \cdot h - \sum_h o_{j,t,h} \cdot h \right)^2. \tag{4}$$

Before solving op2, we need to confirm two things: first, that entities i and j are involved in event e at moment t, and second, that there is a relationship between entities i and j. Here, we introduce two 0-1 constants, $C_{i,e}$ and $B_{e,t}$. $C_{i,e}$ means that entity i is involved in event e. If entity i is involved in event e, then $C_{i,e} = 1$. $B_{e,t}$ means that it is event e that occurs at moment t. If event e occurs at moment t, then $B_{e,t} = 1$. Mathematically, op2 can be expressed as:

$$\min_o \quad \sum_{i,j,t,e} \left(\sum_h o_{i,t,h} \cdot h - \sum_h o_{j,t,h} \cdot h \right)^2 B_{e,t} C_{i,e} C_{j,e} \tag{5}$$

■ **MOO model.** The MOO model is composed of the above two optimization objectives: *line crossing* (Equation (3)) and *distance between entities* (Equation (5)). We utilize a weighted linear combination to convert the above two objective functions as follows:

$$\min_{p,o} \quad \alpha \sum_{i,j,t} p_{i,j,t} p_{j,i,t+1} + \beta \sum_{i,j,t,e} \left(\sum_h o_{i,t,h} \cdot h - \sum_h o_{j,t,h} \cdot h \right)^2 B_{e,t} C_{i,e} C_{j,e}, \tag{6}$$

where α and β represent the weight coefficients utilized for the purpose of harmonizing the two objective functions. After several experiments, we believe that it is more important to reduce line crossings; therefore, we set the values of α as 1 and β as 0.1.

■ **Constraints.** For the problem of identifying the position of entity i in entity sequence at time t, we can transform it into an "*Assignment Problem*", i.e., suppose that at time t we have an order of H boxes and entity i must and can only enter one box h. It can be expressed as

$$\sum_i o_{i,t,h} = 1, \qquad \forall t, h \tag{7a}$$

$$\sum_h o_{i,t,h} = 1, \qquad \forall i, t. \tag{7b}$$

Equation (7)(a) ensures that at any moment t and position h, there is one and only one entity i. Equation (7)(b) ensures that at any entity i and moment t, there is one and only one position h.

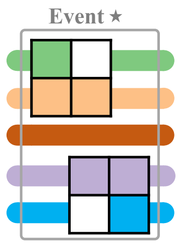


Fig. 6. Events ★ are represented by two separate pixel grids.

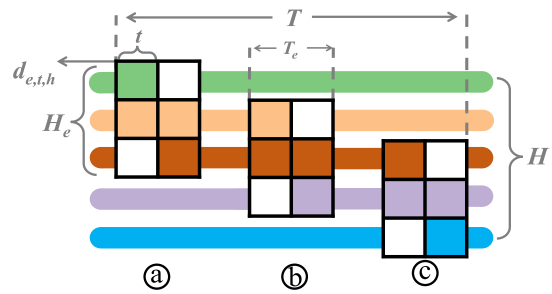


Fig. 7. Diagram of "*Sliding Window*". The state of the window during sliding: (a) at the start of the entity sequence, (b) in the entity sequence, and (c) at the end of the entity sequence.

The mathematical relationship between $o_{i,t,h}$ and $p_{i,j,t}$ is expressed as

$$p_{i,j,t} = 1, \qquad when \quad o_{i,t,h_1} + o_{j,t,h_2} = 2 \;\&\; h_1 < h_2 \tag{8a}$$

$$p_{i,j,t} + p_{j,i,t} = 1, \qquad \forall t, i, j \;\&\; i \neq j \tag{8b}$$

$$p_{i,j,t} = 0, \qquad \forall t, i, j \;\&\; i = j. \tag{8c}$$

In solving, Equation (8)(b) and 8(c) ensure that there is a difference in position between entities i and j at the same moment t, while excluding the case in which only one entity is taken.

In order to achieve op2, while determining the entity positions, we should also attach constraints on the entities involved in the same event e. As mentioned in CDR1, during the duration of event e, the y-axis coordinates of entity i remain unchanged. Here, we define T_e to denote the duration of event e. Its mathematical expression is:

$$o_{i,t_1,h} = o_{i,t_2,h}, \qquad when \quad C_{i,e} = B_{e,t_1} = B_{e,t_2} = 1 \;\&\; t_1, t_2 \in T_e. \tag{9}$$

After experiments, we found that the above constraints alone are not sufficient. In CDR1, we propose to present events in an intuitive way. In CDR2, we expect entities that belong to the same SPO triplet to be close to each other. Unfortunately, the error shown in Figure 6 sometimes arises. The gray box represents the Event ★. We can see from it that an unrelated entity (the brown line) exists in the gray box, which divides the Event ★ into two parts. Therefore, we need to add constraints to avoid mixing irrelevant entity in the group of entities involved in the same event e. We compare the group of entities to a window of length H_e and compare its position adjustment to the sliding of a window, i.e., bundling entities to move together. We define $d_{e,t,h}$, which is a 0-1 decision variable that determines the starting point of the window consisting of entities involved in event e at moment t. During window sliding, there are several situations: the window starting at the beginning of the entity order, the window being within the entity order, and the window ending at the end of the entity order, as illustrated in Figures 7(a), 7(b), and 7(c), which correspond to the three situations, respectively.

Mathematically, Equation (10) describes the "*Sliding Window*". "*Sliding Window*" needs to solve the overflow problem and avoid the entity sequence into a circle. For this purpose, we have to limit the position of the window and ensure that there are no irrelevant entities inside it. First, we need to determine the starting point of the window. Equation (10)(a) ensures that there exists a position for the head of the window when $h \in [1,\; H - H_e + 1]$. Equation (10)(b) ensures that there is no position for the head of the window when $h \in [H - H_e + +2, H]$. Equation (10)(a) and 10(b) together guarantee that there is one and only one point as the head of the window when $h \in [1, H]$. Second, in order to have no irrelevant entities inside the window, we need to bind all the entities involved in event e together. Equation (10)(c) ensures that the entity i involved in the event e cannot be

Table 3. Results of Numerical Experiments

Entity	Event	SPO triplet	Constraint	Time (second)
7	6	40	37,461	214
8	3	12	18,709	112
	8	38	64,310	10,969
	8	50	84,375	11,363
9	3	19	52,224	1,404
	8	56	153,593	43,200

ranked in front of the starting point h. Equation (10)(d) can ensure that the entities involved in the event e are within the window.

$$\sum_{h=1}^{H-H_e+1} d_{e,t,h} = 1, \qquad \forall e,t \;\; when \;\; B_{e,t} = 1, \tag{10a}$$

$$\sum_{h=H-H_e+2}^{H} d_{e,t,h} = 0, \qquad \forall e,t \;\; when \;\; B_{e,t} = 1, \tag{10b}$$

$$\sum_{i=1}^{H} \sum_{h_b=1}^{h} o_{i,t,h_b} C_{i,e} = 0, \qquad \forall e,t \;\; when \;\; d_{e,t,h} = 1, \tag{10c}$$

$$\sum_{i=1}^{H} \sum_{h_e=h+1}^{h+H_e} o_{i,t,h_e} C_{i,e} = H_e, \qquad \forall e,t \;\; when \;\; d_{e,t,h} = 1. \tag{10d}$$

■ **Solving.** The environment for computing our layouts is a Windows system based on an Intel(R) Xeon(R) CPU with 3.40 GHz and 8 GB RAM. Our implementation is a Python program that generates the MOO formulation and solves it using the commercial optimizer *Gurobi* [71]. Limited by the solving efficiency of current solvers, we have to evaluate the computational cost in a comprehensive way when the data volume is large. Gurobi can get the optimal solution of the MOO model by exploring the solution set space. We set the minimum running time of the solving program (30 min). Although the optimal solution is what we seek, the time cost increases rapidly as the solution set space converges to the optimum. Therefore, we also set two conditions for terminating the solving program if the run time exceeds 30 minutes. One is that the absolute gap between the best known solution and boundary is below a desired threshold ($\leq 2\%$). The other is to set a time frame for computing the layouts ($\leq 12h$). Table 3 presents the results of numerical experiments. In most cases, Gurobi generates intermediate solutions quickly, whereas most of the computation time is spent on finding minor improvements to the objective function. In practice, it is also worth examining suboptimal solutions, as in some cases the suboptimal layout may actually be visually more pleasing than the optimal one.

5 RELATIONSHIP DISPLAY

In this section, we will introduce the visual design of relationships between entities. Firstly, it is crucial to effectively demonstrate various types of the entity–entity relationships. To achieve this, we conducted a brainstorming session within a certain constrained framework to explore the design of visual elements that could represent the entity–entity relationships. After thorough discussions, we collected 15 design schemes. Finally, a user study was conducted to gather evaluations of these schemes, which were used as the basis for selecting the final scheme.

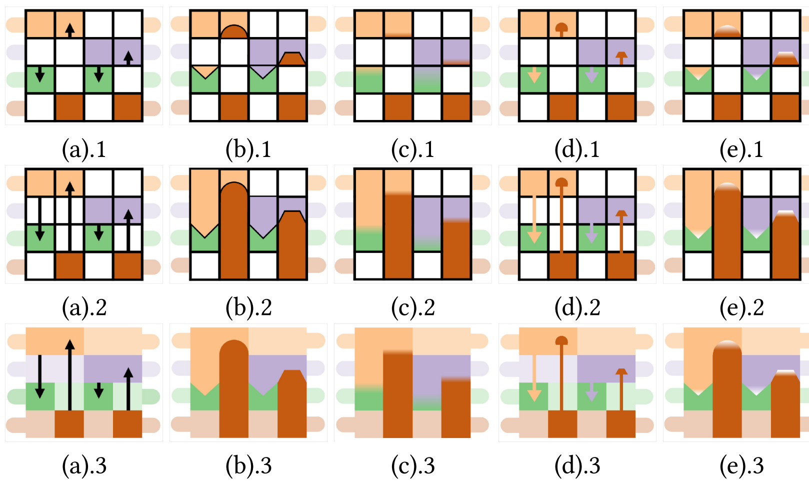


Fig. 8. Design schemes. (a) Directed arrow. (b) Basic shape. (c) Opacity fade. (d) Directed arrow + basic shape. (e) Opacity fade + basic shape.

5.1 Design Schemes

In addition to a well-organized layout, the visual representation of entity–entity relationships is also crucial. Based on emotional categorization or other classification methods, we can roughly divide the categories of entity–entity relationships into several types [72]. For instance, these relationships can be categorized based on emotions (friendly, malicious, neutral) or specific actions (offense, defense, goal). The determination of the appropriate visual representations that can embed relationship types within storyline visualization necessitates validation through user study.

We invited three master students and one PhD student in the field of visualization to conduct a brainstorming session on the topic of "Visual representations of the entity–entity relationships". During the session, we collected several design schemes. After summarizing them, we extracted the following three legends: directed arrow, basic shape, and opacity fade. These legends have directional identifiers, i.e., the "Agent->Patient" in the SPO triplets that we expect to express. Figure 8 shows 15 design schemes for displaying the entity–entity relationships based on the above legends.

① Directed arrow: " ".
② Basic shape: " ", " ", " ".
③ Opacity fade: " ".

In fact, the combination between the above legends forms a good effect as well.

④ Redesign the base shapes into directed arrows: " ", " ".
⑤ Redesign the base shapes with opacity fade: " ", " ", " ".

5.2 User Study

5.2.1 User Study for Design Schemes. To assess the above design schemes, we investigated 24 participants by means of a questionnaire. Four of the participants are undergraduate students and the rest are graduate students. In the questionnaire, we conveyed to the participants the design goals and evaluation principles. Then, we showed each participant the five groups of design schemes in Figure 8 separately. Participants chose their favorite design scheme in each group based on individual subjectivity. Thereafter, the participants progressed further to choose the final one

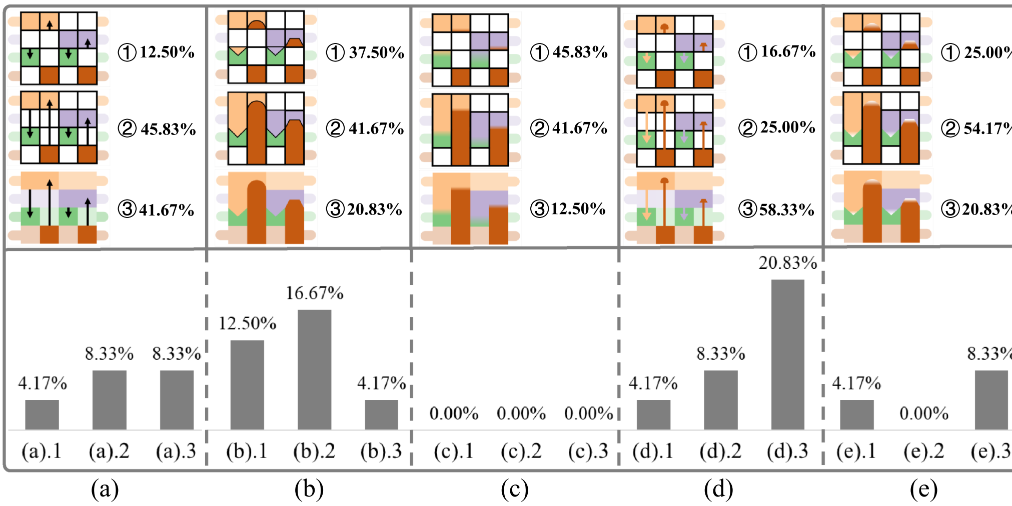


Fig. 9. Design scheme statistics. The statistical results of each group of design schemes are visual mapped with a histogram.

among the schemes selected from each group. At the end of the questionnaire, we asked the participants to provide the reasons for their choices. The statistical results are shown in Figure 9.

The top half of Figure 9 shows schemes for each group and the percentage of being selected. The bottom half shows the statistics of participants' final choices. We found several interesting things in it. Firstly, scheme c "opacity fade" is the most unpopular. Along with it, scheme e "opacity fade + basic shape" also becomes less attractive. Secondly, scheme b and scheme d have a higher percentage of participants' final choices. Thirdly, scheme e.2 has the highest percentage of choices in the scheme e group, but it is not the final choice of participants. Fourthly, scheme d.3 is the one with the highest percentage of choices in the group of scheme d, and it is also the most accessible one.

Here, we have randomly selected three participants to show their choices and reasons. **Participant 1** is an undergraduate student with no visualization background. In the questionnaire, she chose schemes a.1, b.3, c.1, d.3, and e.2; her final choice was scheme d.3. In her opinion, the black border would create a separation problem. Thus, she chose the schemes of short arrows and the schemes that eliminate the black border, as the "opacity fade" was not aesthetically pleasing. **Participant 2** is an MS student in graphic design. She chose schemes a.3, b.3, c.3, d.3, and e.3 in the questionnaire. She believed that black borders would be visually constricting and not suitable for expressing connections. Thus, she chose all options without black borders. Schemes b.3 and e.3 made the whole design tend to be one, with the feeling of a whole, whereas scheme c.3 had a sense of confusion. She decided to choose scheme d.3 because of its clear directionality and the easy association of "they are related to each other". **Participant 3** is a PhD student in the field of visualization. For each of the five schemes in the questionnaire, she chose schemes a.2, b.2, c.3, d.1, and e.2. As far as she was concerned, it was crucial to clearly distinguish between *Agent* and *Patient* black borders enabled this function. In addition, she considered that visual coherence was very important and that *Agent* and *Patient* should be connected as directly as possible. In the end, she chose scheme b.2, which gave her quick access to the semantic relationships of the entities while bringing her minimal visual clutter and visual interference.

From the statistics and participant interviews, we found that most of participants had little preference for schemes with black borders. They felt that black borders would create a sense of separation. This situation is reversed, nevertheless, in the case of schemes with only base shapes,

Table 4. Examined Schemes using *Gestalt Theory*

Gestalt Theory / Schemes	Proximity	Similarity	Continuity	Closure	Common fate	Good figure	Symmetry	Past experience
Scheme d.3	✓	✓	✓	✓	✓	✓	✓	✓
Scheme b.2	✓	✓	✓		✓	✓	✓	✓
Scheme c.1							✓	
Scheme e.2	✓	✓	✓		✓		✓	✓

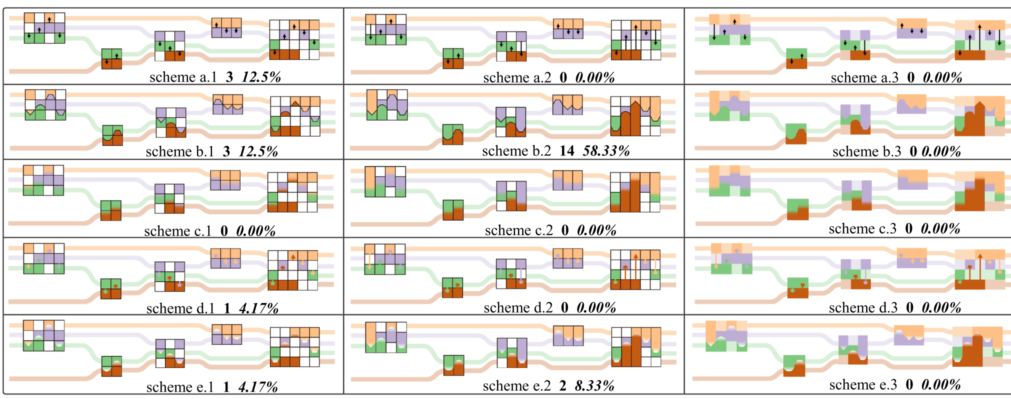


Fig. 10. Illustration of all schemes embedded in storyline visualization and their statistical results.

since having only the base shape would make schemes too monolithic to require proper separation. *Gestalt theory* [73, 74] can be used to explain this phenomenon. Gestalt is a psychological movement in which the important idea is that people's perceptions of objective phenomena originate from overall relationships rather than specific elements. The whole precedes the elements, and the nature of local elements is determined by the structural relationships of the whole. This is because human perception of any visual image is organized by perceptual system in terms of form and contour. Gestalt theory includes proximity, similarity, continuity, closure, common fate, good figure, symmetry, and past experience. In Table 4, we examined the two most favored schemes (schemes d.3 and b.2) and the two least favored schemes (schemes c.1 and e.2) through *Gestalt theory*. By comparing these four schemes, we find that closure has a negative impact, but its effect is minor, whereas whether the scheme has good figure has a significant impact.

5.2.2 Embedding Design Schemes into Storyline Visualization. In this section, we embed all design schemes into storyline visualization, as shown in Figure 10. We first randomly generated 5 events and 17 SPO triplets, and proceeded to use them as input data for an optimal solution with the MOO model. Interestingly, after embedding design schemes into storyline visualization, the more user-friendly design scheme changed. More than half of the participants preferred the results of embedding scheme b.2 into storyline visualization, followed by schemes b.1 and a.1.

We interviewed the previous three participants again. **Participant 1** changed her choice from scheme d.3 to scheme b.2. She felt that scheme d.3, when embedded in storyline visualization, would confuse her, because the distinction was not clear enough. For scheme b.1, she still believed that too many partitions would affect the aesthetics. **Participant 2** insisted that a black border would be visually burdensome to her, so she did not change her choice, nor did **Participant 3**. She commented, "*While all three of these provide me with a quick overview of the number of entities,*

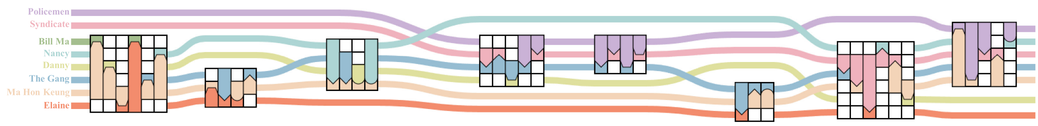


Fig. 11. Illustrating *Rumble in the Bronx* with E^2Storyline.

the different events with involved entities, and the semantic relationships between entities, the white space in scheme b.1 is quite visually burdensome for me, but scheme b.2 eliminates the feeling of 'fragmentation'. Scheme d.3 has arrows with different shapes, which also strikes me as not simple enough, and its straight through the middle entity is a bit abrupt." After a comprehensive analysis of two user studies, we decided to use scheme b.2.

6 EVALUATION

In this section, we evaluate the usability and effectiveness of E^2Storyline with several cases of text data. Furthermore, we compare E^2Storyline with other forms of storyline visualization.

6.1 Text Synopsis of a Movie

Figure 11 illustrates the story evolution of the movie *Rumble in the Bronx*. It can be seen that the story consists of several main events, and the entities involved in the same event are well brought together. We classify the entity–entity relationships into three categories: neutral "⏢", friendly "◓", and malicious "▲". Based on this movie synopsis, we evaluated the intuitiveness and effectiveness of E^2Storyline through a user study. We took the task framework of Brehmer et al.'s multi-level typology [75] into consideration and designed the following seven tasks. Tasks 1–4 are designed based on events and Tasks 5–7 are designed based on characters. For the answer to each task, we give a ground truth. The task details are shown in the table in Figure 12. User tasks are as follows:

- **Task 1:** Please answer the number of events included in E^2Storyline.
- **Task 2:** Please select the event with the most participating characters.
- **Task 3:** Please select the event with the least participating characters.
- **Task 4:** Please select the event with the most complex entity relationships.
- **Task 5:** Please choose the two most active characters.
- **Task 6:** Please select the head villain and count the number of malicious behaviors he was involved in.
- **Task 7:** Please select the friendliest character and count the number of friendly behaviors he was involved in.

We recruited 14 participants (8 males and 6 females, ranging in age from 20 to 27) through a student email list. Of these, 3 participants were undergraduate students and 11 participants were postgraduate. Six participants possessed practical experience in data analysis, three were actively involved in computer vision–related research, and three had a comprehensive understanding of data visualization. The other two had limited technical backgrounds but had a strong interest in visual analysis, and had studied data visualization, web design, human–computer interaction, and other courses. In addition to knowing the personal information of the participants, we also ensured that all participants had no visual cognitive impairment in terms of color blindness or color weakness, and had no difficulty in understanding visual mapping. A laptop computer was used, with a 25-inch display of resolution 1920 × 1080 pixels.

First, each participant started with a tutorial lasting approximately 5 minutes introducing the E^2Storyline, data context, and user tasks. Second, we ensured that each participant completed

Rumble in the Bronx

Task Category		Ground Truth
Event-based	Task 1	8 events
	Task 2	Event #1 or #7
	Task 3	Event #2 or #5 or #6
	Task 4	Event #8
Entity-based	Task 5	Ma Hun Keung; The Gang
	Task 6	Syndicate; 11
	Task 7	Ma Hun Keung; 4

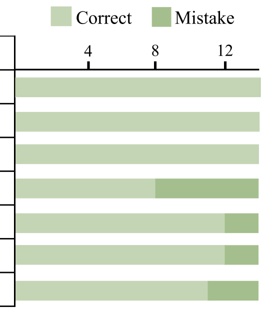


Fig. 12. 7 user tasks, ground truth, and statistical results of *Rumble in the Bronx*.

user tasks independently. Only one assessment facilitator and one assessment recorder were allowed to be present during the participant's task performance, and the two staff were not allowed to interfere with the participant's task performance. Then, we asked the participant to perform user tasks and give personal insights based on evaluation indicators. The execution process of user tasks and the interview content were recorded in real time by the assessment recorder. We collated the feedback results for each user task, and counted the number of people who answered correctly, mistakenly, and ambiguously. The statistical results are presented in the stack diagram in Figure 12.

In the event-based user tasks, we found that the participants paid more attention to the gathering area of the event during the analysis process while ignoring the information of the character itself. For example, in Task 1, all participants found exactly eight events. In addition, eight participants held that the most complex events were Events ♯1 and ♯7. They only focused on the number of entities involved and the space occupied by the pixel grid, but they ignored the types of relationships between entities. In fact, there are more types of the entity–entity relationships in Event ♯8 than in Events ♯1 and ♯7.

In the entity-based user tasks, we observed that a majority of participants demonstrated the ability to distinguish characters in the movie and comprehend their role positioning. This was achieved by analyzing the relationship and behavior types between characters, utilizing the visual design of E^2Storyline. A few participants were not familiar enough with our behavioral type design scheme, leading to errors in their analysis, especially in counting the number of behaviors. A few participants were ambiguous about the results.

We also collated participant feedback on the interpretability and usability of E^2Storyline. About 71.5% of the participants held that E^2Storyline was easy to understand, and about 85% of the participants believed that E^2Storyline helped to understand movies, events, and character relationships. Three participants volunteered to ask us whether Event ♯4 was a turning event in the story, as it included two characters who had not previously been involved in any events. It shows that E^2Storyline can help users discover the function of events in the whole movie. Many participants observed that "malicious" relationships were the most frequent of the three genres. Therefore, they believed that the movie was an action movie with a high probability. In fact, its genre is indeed an action movie. This shows that E^2Storyline can help users to determine the movie genre to some extent.

We also made a presentation of *The Matrix* with E^2Storyline, as shown in Figure 13, and asked the participants how they felt after observing the storyline visualization. Some participants believed that the character "Morpheus" was a good guy. This is because they found that most of the types of relationships involving him were friendly. This suggests that E^2Storyline can help users determine the identity of entities.

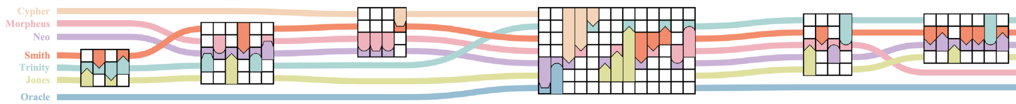


Fig. 13. Illustrating *The Matrix* with E^2Storyline.

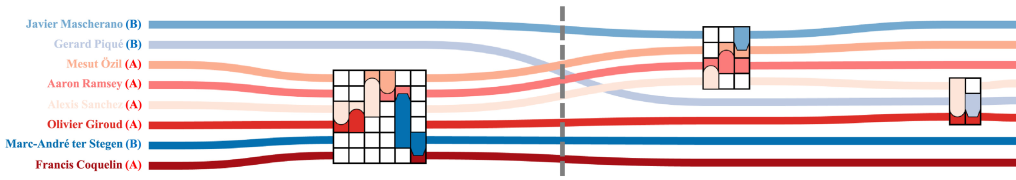


Fig. 14. Arsenal's midfield and front players against Barcelona's midfield and back players.

6.2 Live Text Commentary of a Football Match

Live text commentary is a good alternative when it is not convenient to watch live matches and news in video form. To meet the immediacy of reporting, the sentences in live text data are characterized by brevity but also preserve the general structure of sentences. Therefore, we can extract structured SPO triplets from the data. Moreover, there are almost no time jumps in live text commentary; thus, it is highly linear in time.

The post-game highlights videos were intercepted by professionals; thus, we selected events based on them. We visualized this game with E^2Storyline, as shown in Figures 14 and 15. In these two diagrams, "◖◗" stands for offense, "▰" for defense, "▲" for goals, and gray dotted lines for halftime. The players' names on the left side of the diagram are represented by "A" for Arsenal and colored in red, and by "B" for Barcelona and colored in blue [76]. We analyzed this game from 2 perspectives: ❶ Arsenal's midfield and front players against Barcelona's midfield and back players, as shown in Figure 14; and ❷ Barcelona's midfield and front players against Arsenal's midfield and back players, as shown in Figure 15. We got some conclusions from each of these two perspectives, which gave us a more comprehensive understanding of this match.

Perspective ❶: The following information can be obtained from Figure 14. First, Arsenal's most active offense occurred in the first half. Second, among Arsenal's attacking players, Sanchez was the most active player. Third, most of Arsenal's attacks were initiated by Sanchez (left winger) and passed to Giroud (center forward), whereas Barcelona's defenders were Mascherano and Piqué (center backs), indicating that Arsenal's tactics were mostly frontal attacks. This is in line with Arsenal's "4-2-3-1" formation. The diagram hints that Arsenal's scoring difficulties might be because Barcelona has a strong defense given that Arsenal had more attackers than Barcelona had defenders.

Perspective ❷: The following information can be obtained from Figure 15. First, Barcelona was more active in the second half than in the first half. Second, Barcelona's players had more interaction with each other, especially Suarez, Neymar, and Messi. Third, the two attacks that led to goal were similar —Barcelona's midfielder passing to wingers. The two wingers, Neymar and Suarez, drew the Arsenal defenders and finally passed the ball to Messi for goals.

6.3 Comparison

In this section, we compared E^2Storyline with existing visualization techniques for storyline visualization and dynamic graph. The following three diagrams are drawn using *Storyflow* (Figure 16(a)) [12], node link–based dynamic graph (Figure 16(b)) [77], and E^2Storyline (Figure 16(c)). In *StoryFlow*, Liu et al. grouped entities appearing in the same location and sorted them using a

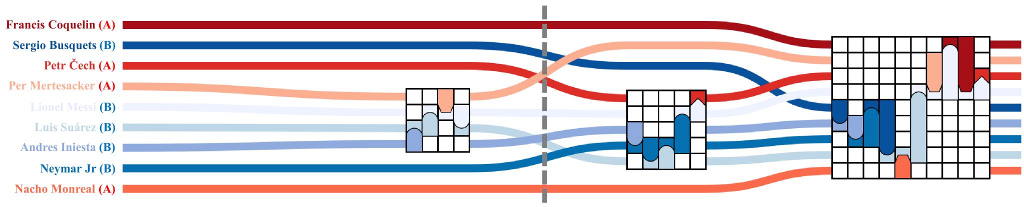
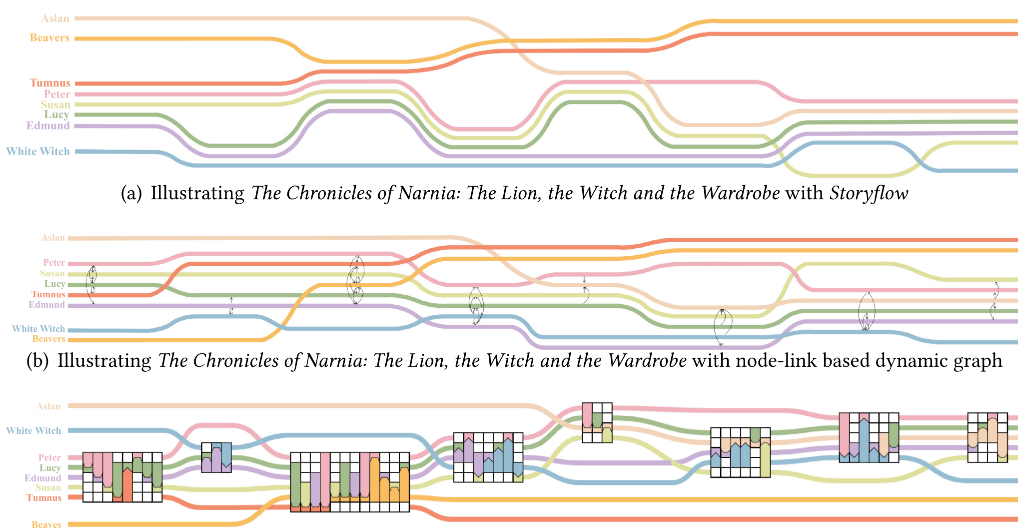


Fig. 15. Barcelona's midfield and front players against Arsenal's midfield and back players.



(a) Illustrating *The Chronicles of Narnia: The Lion, the Witch and the Wardrobe* with *Storyflow*

(b) Illustrating *The Chronicles of Narnia: The Lion, the Witch and the Wardrobe* with node-link based dynamic graph

(c) Illustrating *The Chronicles of Narnia: The Lion, the Witch and the Wardrobe* with E^2Storyline

Fig. 16. Illustrating *The Chronicles of Narnia: The Lion, the Witch and the Wardrobe* with (a) *StoryFlow*, (b) node-link based dynamic graph, and (c) E^2Storyline.

directed acyclic graph (DAG) method to obtain an optimized storyline. Here, we change the basis for grouping entities from "location" to "event". In a *node link–based dynamic graph*, the grouping is also based on "events" but with the addition of the entity–entity relationships. In contrast to *Storyflow*, it needs to consider the constraints of the entity–entity relationships when optimizing the layout. These three diagrams, although different in form, show the same movie, *The Chronicles of Narnia: The Lion, the Witch and the Wardrobe*. We propose several hypotheses. **H1**: *StoryFlow* is not capable of expressing the relationships between entities. **H2**: a *node link–based dynamic graph* fails to show the temporal information of the entity–entity relationships in detail. **H3**: neither *Storyflow* nor a *node link–based dynamic graph* allows users to gain insight into stories, such as finding a certain type of behavior quickly and accurately or being specific to a key behavior of an entity. To test these hypotheses, we consulted two PhD students familiar with dynamic graphs and temporal data in the form of interviews in three dimensions: entity, property, and temporal features [78].

Task 1: Locate specific characters (e.g., Lucy or White Witch).
Task 2: Locate specific behaviors (e.g., Edmund's first negative behavior or Lucy's last friendly behavior).
Task 3: Retell special events (e.g., Event ♯7 in *The Chronicles of Narnia*).

In response to Task 1, two PhD students agreed that all three storyline visualizations made it easy to find characters in the story. In response to Task 2, they argued that drawing with *StoryFlow* cannot perform the task of finding specific behaviors. This confirms H1 and H3, that *StoryFlow* fails to convey information about the entity–entity relationships to users. While a "node link–based dynamic graph is difficult to find accurately, despite the fact that it can show the entity–entity relationships" is in accordance with our conjecture of a node link–based dynamic graph in H3, they agreed that the one drawn with E^2Storyline could easily find the desired entity or relationship. It shows more of the story while taking into account the advantages of the other two formats. They consider Task 3 to be a progression of Task 2. Fortunately, the ones drawn with E^2Storyline can still handle it to some extent. They roughly stated what happened based on the order of the relationships between entities in Event ♯7, *"Peter and Edmund attacked White Witch, but were met with a counterattack from White Witch. However, they couldn't beat White Witch and Edmund was even injured. Eventually Aslan appeared to defeat White Witch and Lucy treated Edmund."* They argue that the reason why node link–based dynamic graphs cannot help users retell the evolution of stories is because they are unable to show the temporal information of the entity–entity relationships. This is in line with what we proposed in H2. Moreover, as events become complex, the lines connecting nodes inevitably cross, e.g., the third event in Figure 16(b).

7 DISCUSSION AND FUTURE WORK

In this section, we review the entire workflow, discuss the advantages and limitations of E^2Storyline in a comprehensive manner based on case studies and user feedback, and reflect on future work.

We believe that it is not enough to reflect story trends in storyline visualization. The entity–entity relationships in events are equally important. To do this, we propose E^2Storyline. We extract SPO triplets from textual data and use them to describe the relationships between entities. We followed the accepted visual design principles for storyline and also came up with some custom design requirements. We described the layout problem of storyline using MOO and solved it with *Gurobi*. For the visual representation of the entity–entity relationships, we brainstormed and did user studies on different design schemes. We made case studies with movie text synopses and text live commentaries as well as making a comparison with *StoryFlow* and *node link–based dynamic graphs*. Overall, we found that participants using E^2Storyline could quickly and accurately find important events in the story and could further analyze the role of events and some patterns of entities. In the future, there is still research to be done on an ongoing basis.

Algorithm. Layout optimization requires the integration of multiple design requirements; thus, we formulate it as a MOO problem. It can be solved using Gurobi to obtain the global optimum, which ensures that the layout is the best fit to the design requirements. As the size of the data increases, limited by the efficiency of Gurobi, solving the global optimum will cost more time. This makes the algorithm poor in real time. To overcome the poor real-time performance, in future work, we will make the following attempts: ❶ Develop rules for data preprocessing aimed at size reduction. ❷ Employ a synergy of algorithms, including multi-objective optimization and genetic algorithms. ❸ Apply ML techniques to address layout optimization challenges.

Visual design. In the realm of visual design, we employed brainstorming sessions and conducted user studies to identify a visual design framework for illustrating entity relationships. This design framework exhibits commendable directionality, effectively conveying action types to users without introducing unnecessary complexity. The efficacy of this design framework is evident from the results of our case studies. However, due to the inherent diversity of base graphics, it falls short in expressing a wide array of relationship types. To address this limitation and expand the visual representation of relationships, our subsequent efforts will involve exploring alternative visual design frameworks and implementing interactive features such as highlighting and boxing. Additionally, in the user study of our visual scheme, we did not adequately account for the demographics of the participants (such as age, education level, and knowledge background). We plan to conduct more detailed and comprehensive user studies and case studies in the future. These endeavors will involve a broader range of demographics and will be compared against a greater variety of storyline visualizations to validate the advantages of the design proposed in this article.

Overall, storyline visualization is useful and effective as a method of providing an overview of story trends. There remains value in future research in glyph design and efficient layout algorithms.

8 CONCLUSION

We proposed a novel method of storyline layout that focuses on showing the relationships between entities in the various events of the story. We performed a tailor-made requirements analysis for both data and visual design, and described the storyline layout with the MOO problem mathematically. We cite several case studies and conduct user studies. The results show that our participants can quickly find the events in the story when using E^2Storyline. Participants were also able to develop an efficient and effective analysis of the roles of events and relationships between entities.

ACKNOWLEDGMENTS

The authors thank all participants of user study and reviewers for their valuable comments.

REFERENCES

[1] Robert Kosara and Jock Mackinlay. 2013. Storytelling: The next step for visualization. *Computer* 46, 5 (2013), 44–50.
[2] Lu Qiang, Chai Bingjie, and Zhang Haibo. 2017. Storytelling by the StoryCake visualization. *The Visual Computer* 33, 10 (2017), 1241–1252.
[3] Jänicke Stefan, Greta Franzini, Muhammad Faisal Cheema, and Scheuermann Gerik. 2015. On close and distant reading in digital humanities: A survey and future challenges. *EuroVis (STARs)* 2015 (0215), 83–103.
[4] Panpan Xu, Yingcai Wu, Enxun Wei, Tai-Quan Peng, Shixia Liu, Jonathan J. H. Zhu, and Huamin Qu. 2013. Visual analysis of topic competition on social media. *IEEE Transactions on Visualization and Computer Graphics* 19, 12 (2013), 2012–2021.
[5] Guodao Sun, Yingcai Wu, Shixia Liu, Tai-Quan Peng, Jonathan J. H. Zhu, and Ronghua Liang. 2014. EvoRiver: Visual analysis of topic coopetition on social media. *IEEE Transactions on Visualization and Computer Graphics* 20, 12 (2014), 1753–1762.
[6] Jeffery Ansah, Lin Liu, Wei Kang, Selasie Kwashie, Jixue Li, and Jiuyong Li. 2019. A graph is worth a thousand words: Telling event stories using timeline summarization graphs. In *The World Wide Web Conference*. 2565–2571.
[7] Bang Liu, Fred X. Han, Di Niu, Linglong Kong, Kunfeng Lai, and Yu Xu. 2020. Story Forest: Extracting events and telling stories from breaking news. *ACM Transactions on Knowledge Discovery from Data* 14, 3 (2020), 1–28.
[8] Ayush Shrestha, Ben Miller, Ying Zhu, and Yi Zhao. 2013. Storygraph: Extracting patterns from spatio-temporal data. In *Proceedings of the ACM SIGKDD Workshop on Interactive Data Exploration and Analytics*. 95–103.
[9] Xi He and Ying Zhu. 2020. Integrated spatio-temporal storyline visualization with low crossover. In *International Conference Information Visualisation (IV'20)*. IEEE, 236–241.
[10] R. Munroe. ([n. d.]) Movie Narrative Charts. https://xkcd.com/657/. ([n. d.]).
[11] Yuzuru Tanahashi and Kwan-Liu Ma. 2012. Design considerations for optimizing storyline visualizations. *IEEE Transactions on Visualization and Computer Graphics* 18, 12 (2012), 2679–2688.
[12] Shixia Liu, Yingcai Wu, Enxun Wei, Mengchen Liu, and Yang Liu. 2013. StoryFlow: Tracking the evolution of stories. *IEEE Transactions on Visualization and Computer Graphics* 19, 12 (2013), 2436–2445.
[13] Tan Tang, Sadia Rubab, Jiewen Lai, Weiwei Cui, Lingyun Yu, and Yingcai Wu. 2018. iStoryline: Effective convergence to hand-drawn storylines. *IEEE Transactions on Visualization and Computer Graphics* 25, 1 (2018), 769–778.
[14] Makarand Tapaswi, Martin Bauml, and Rainer Stiefelhagen. 2014. StoryGraphs: Visualizing character interactions as a timeline. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 827–834.
[15] Zihao Zhu, Yue Shen, Sujia Zhu, Gefei Zhang, Ronghua Liang, and Guodao Sun. 2022. Towards better pattern enhancement in temporal evolving set visualization. *Journal of Visualization* (2022), 1–1.

[16] Tuan Nhon Dang, Nick Pendar, and Angus Graeme Forbes. 2016. TimeArcs: Visualizing fluctuations in dynamic networks. In *Computer Graphics Forum*, Vol. 35. Wiley Online Library, 61–69.

[17] Nathalie Henry, Jean-Daniel Fekete, and Michael J. McGuffin. 2007. Nodetrix: A hybrid visualization of social networks. *IEEE Transactions on Visualization and Computer Graphics* 13, 6 (2007), 1302–1309.

[18] Daniela Oelke, Dimitrios Kokkinakis, and Daniel A. Keim. 2013. Fingerprint matrices: Uncovering the dynamics of social networks in prose literature. In *Computer Graphics Forum*, Vol. 32. Wiley Online Library, 371–380.

[19] Jian Zhao, Michael Glueck, Fanny Chevalier, Yanhong Wu, and Azam Khan. 2016. Egocentric analysis of dynamic networks with EgoLines. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. 5003–5014.

[20] Michael Ogawa and Kwan-Liu Ma. 2010. Software evolution storylines. In *Proceedings of the International Symposium on Software Visualization*. 35–42.

[21] Yang Shi, Chris Bryan, Sridatt Bhamidipati, Ying Zhao, Yaoxue Zhang, and Kwan-Liu Ma. 2018. MeetingVis: Visual narratives to assist in recalling meeting context and content. *IEEE Transactions on Visualization and Computer Graphics* 24, 6 (2018), 1918–1929.

[22] Qiang Lu, Xiang-yuan Zhu, Li Liu, and Shu-bo Cao. 2014. An effective demonstration for group collaboration based on storyline visualization technology. In *Proceedings of the International Conference on Computer Supported Cooperative Work in Design*. IEEE, 47–52.

[23] Tan Tang, Renzhong Li, Xinke Wu, Shuhan Liu, Johannes Knittel, Steffen Koch, Thomas Ertl, Lingyun Yu, Peiran Ren, and Yingcai Wu. 2020. PlotThread: Creating expressive storyline visualizations using reinforcement learning. *IEEE Transactions on Visualization and Computer Graphics* 27, 2 (2020), 294–303.

[24] Fabian Bolte, Mahsan Nourani, Eric D. Ragan, and Stefan Bruckner. 2020. SplitStreams: A visual metaphor for evolving hierarchies. *IEEE Transactions on Visualization and Computer Graphics* 27, 8 (2020), 3571–3584.

[25] Wei Zhang, Siwei Tan, Siming Chen, Linhao Meng, Tianye Zhang, Rongchen Zhu, and Wei Chen. 2022. Visual reasoning for uncertainty in spatio-temporal events of historical figures. *IEEE Transactions on Visualization and Computer Graphics* (2022).

[26] Yuzuru Tanahashi, Chien-Hsin Hsueh, and Kwan-Liu Ma. 2015. An efficient framework for generating storyline visualizations from streaming data. *IEEE Transactions on Visualization and Computer Graphics* 21, 6 (2015), 730–742.

[27] Martin Gronemann, Michael Jünger, Frauke Liers, and Francesco Mambelli. 2016. Crossing minimization in storyline visualization. In *Proceedings of the International Symposium on Graph Drawing and Network Visualization*. 367–381.

[28] Martin Nollenburg and Alexander Wolff. 2010. Drawing and labeling high-quality metro maps by mixed-integer programming. *IEEE Transactions on Visualization and Computer Graphics* 17, 5 (2010), 626–641.

[29] Ben Jacobsen, Markus Wallinger, Stephen Kobourov, and Martin Nöllenburg. 2020. MetroSets: Visualizing sets as metro maps. *IEEE Transactions on Visualization and Computer Graphics* 27, 2 (2020), 1257–1267.

[30] Yi Guo, Shunan Guo, Zhuochen Jin, Smiti Kaul, David Gotz, and Nan Cao. 2021. A survey on visual analysis of event sequence data. *IEEE Transactions on Visualization and Computer Graphics* (2021).

[31] Chao Tong, Richard Roberts, Rita Borgo, Sean Walton, Robert S. Laramee, Kodzo Wegba, Aidong Lu, Yun Wang, Huamin Qu, and Qiong Luo. 2018. Storytelling and visualization: An extended survey. *Information* 9, 3 (2018), 65.

[32] Nam Wook Kim, Benjamin Bach, Hyejin Im, Sasha Schriber, Markus Gross, and Hanspeter Pfister. 2017. Visualizing nonlinear narratives with story curves. *IEEE Transactions on Visualization and Computer Graphics* 24, 1 (2017), 595–604.

[33] Xingyu Lan, Xinyue Xu, and Nan Cao. 2021. Understanding narrative linearity for telling expressive time-oriented stories. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. 1–13.

[34] Sara Di Bartolomeo, Aditeya Pandey, Aristotelis Leventidis, David Saffo, Uzma Haque Syeda, Elin Carstensdottir, Magy Seif El-Nasr, Michelle A. Borkin, and Cody Dunne. 2020. Evaluating the effect of timeline shape on visualization task performance. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. 1–12.

[35] Catherine Plaisant, Richard Mushlin, Aaron Snyder, Jia Li, Dan Heller, and Ben Shneiderman. 2003. LifeLines: Using visualization to enhance navigation and analysis of patient records. In *Proceedings of the Craft of Information Visualization*. 308–312.

[36] Katie Watson, Samuel S. Sohn, Sasha Schriber, Markus Gross, Carlos Manuel Muniz, and Mubbasir Kapadia. 2019. StoryPrint: An interactive visualization of stories. In *Proceedings of the International Conference on Intelligent User Interfaces*. 303–311.

[37] Benjamin Bach, Conglei Shi, Nicolas Heulot, Tara Madhyastha, Tom Grabowski, and Pierre Dragicevic. 2015. Time curves: Folding time to visualize patterns of temporal evolution in data. *IEEE Transactions on Visualization and Computer Graphics* 22, 1 (2015), 559–568.

[38] Patrick Riehmann, Manfred Hanfler, and Bernd Froehlich. 2005. Interactive Sankey diagrams. In *Proceedings of the IEEE Symposium on Information Visualization*. 233–240.

[39] Krist Wongsuphasawat and David Gotz. 2012. Exploring flow, factors, and outcomes of temporal event sequences with the outflow visualization. *IEEE Transactions on Visualization and Computer Graphics* 18, 12 (2012), 2659–2668.

[40] Krist Wongsuphasawat, John Alexis Guerra Gómez, Catherine Plaisant, Taowei David Wang, Meirav Taieb-Maimon, and Ben Shneiderman. 2011. LifeFlow: Visualizing an overview of event sequences. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 1747–1756.

[41] Megan Monroe, Rongjian Lan, Hanseung Lee, Catherine Plaisant, and Ben Shneiderman. 2013. Temporal event sequence simplification. *IEEE Transactions on Visualization and Computer Graphics* 19, 12 (2013), 2227–2236.

[42] Ji Soo Yi, Niklas Elmqvist, and Seungyoon Lee. 2010. TimeMatrix: Analyzing temporal social networks using interactive matrix-based visualizations. *International Journal of Human–Computer Interaction* 26, 11-12 (2010), 1031–1051.

[43] Benjamin Bach, Emmanuel Pietriga, and Jean-Daniel Fekete. 2014. Visualizing dynamic networks with matrix cubes. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 877–886.

[44] Jian Zhao, Zhicheng Liu, Mira Dontcheva, Aaron Hertzmann, and Alan Wilson. 2015. MatrixWave: Visual comparison of event sequence data. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*. 259–268.

[45] Adam Perer and Fei Wang. 2014. Frequence: Interactive mining and visualization of temporal frequent event sequences. In *Proceedings of the International Conference on Intelligent User Interfaces*. 153–162.

[46] Zhicheng Liu, Bernard Kerr, Mira Dontcheva, Justin Grover, Matthew Hoffman, and Alan Wilson. 2017. CoreFlow: Extracting and visualizing branching patterns from event sequences. *Computer Graphics Forum* 36, 3 (2017), 527–538.

[47] Shunan Guo, Zhuochen Jin, David Gotz, Fan Du, Hongyuan Zha, and Nan Cao. 2018. Visual progression analysis of event sequence data. *IEEE Transactions on Visualization and Computer Graphics* 25, 1 (2018), 417–426.

[48] Tom Baumgartl, Markus Petzold, Marcel Wunderlich, M. Hohn, Daniel Archambault, M. Lieser, A. Dalpke, Simone Scheithauer, Michael Marschollek, and V. M. Eichel. 2020. In search of patient zero: Visual analytics of pathogen transmission pathways in hospitals. *IEEE Transactions on Visualization and Computer Graphics* 27, 2 (2020), 711–721.

[49] Emanuel Zgraggen, Steven M. Drucker, Danyel Fisher, and Robert DeLine. 2015. Visual regular expressions for querying and exploring event sequences. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*. 2683–2692.

[50] Bram C. M. Cappers and Jarke J. van Wijk. 2017. Exploring multivariate event sequences using rules, aggregations, and selections. *IEEE Transactions on Visualization and Computer Graphics* 24, 1 (2017), 532–541.

[51] Jessica Magallanes, Tony Stone, Paul D. Morris, Suzanne Mason, Steven Wood, and Maria-Cruz Villa-Uriol. 2021. Sequen-C: A multilevel overview of temporal event sequences. *IEEE Transactions on Visualization and Computer Graphics* (2021).

[52] Vanessa Peña-Araya, Tong Xue, Emmanuel Pietriga, Laurent Amsaleg, and Anastasia Bezerianos. 2022. HyperStorylines: Interactively untangling dynamic hypergraphs. *Information Visualization* 21, 1 (2022), 38–62.

[53] Siming Chen, Jie Li, Gennady Andrienko, Natalia Andrienko, Yun Wang, Phong H. Nguyen, and Cagatay Turkay. 2018. Supporting story synthesis: Bridging the gap between visual analytics and storytelling. *IEEE Transactions on Visualization and Computer Graphics* 26, 7 (2018), 2499–2516.

[54] Jiang Wu, Dongyu Liu, Ziyang Guo, Qingyang Xu, and Yingcai Wu. 2021. TacticFlow: Visual analytics of ever-changing tactics in racket sports. *IEEE Transactions on Visualization and Computer Graphics* 28, 1 (2021), 835–845.

[55] Weiwei Cui, Shixia Liu, Zhuofeng Wu, and Hao Wei. 2014. How hierarchical topics evolve in large text corpora. *IEEE Transactions on Visualization and Computer Graphics* 20, 12 (2014), 2281–2290.

[56] Samah Gad, Waqas Javed, Sohaib Ghani, Niklas Elmqvist, Tom Ewing, Keith N. Hampton, and Naren Ramakrishnan. 2015. ThemeDelta: Dynamic segmentations over temporal topic models. *IEEE Transactions on Visualization and Computer Graphics* 21, 5 (2015), 672–685.

[57] Corinna Vehlow, Fabian Beck, Patrick Auwärter, and Daniel Weiskopf. 2015. Visualizing the evolution of communities in dynamic graphs. In *Computer Graphics Forum*, Vol. 34. Wiley Online Library, 277–288.

[58] Shunan Guo, Ke Xu, Rongwen Zhao, David Gotz, Hongyuan Zha, and Nan Cao. 2017. EventThread: Visual summarization and stage analysis of event sequence data. *IEEE Transactions on Visualization and Computer Graphics* 24, 1 (2017), 56–65.

[59] Sara Di Bartolomeo, Yixuan Zhang, Fangfang Sheng, and Cody Dunne. 2020. Sequence braiding: Visual overviews of temporal event sequences and attributes. *IEEE Transactions on Visualization and Computer Graphics* 27, 2 (2020), 1353–1363.

[60] Eva Mayr and Florian Windhager. 2018. Once upon a spacetime: Visual storytelling in cognitive and geotemporal information spaces. *ISPRS International Journal of Geo-Information* 7, 3 (2018), 96.

[61] Renaud Blanch, Rémy Dautriche, and Gilles Bisson. 2015. Dendrogramix: A hybrid tree-matrix visualization technique to support interactive exploration of dendrograms. In *2015 IEEE Pacific Visualization Symposium (PacificVis'15)*. IEEE, 31–38.

[62] Zhicheng Liu, Bernard Kerr, Mira Dontcheva, Justin Grover, Matthew Hoffman, and Alan Wilson. 2017. CoreFlow: Extracting and visualizing branching patterns from event sequences. In *Computer Graphics Forum*, Vol. 36. Wiley Online Library, 527–538.

[63] Johannes Hoffart, Fabian M. Suchanek, Klaus Berberich, and Gerhard Weikum. 2013. YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia. *Artificial Intelligence* 194 (2013), 28–61.
[64] Dustin Arendt and Meg Pirrung. 2017. The "y" of it matters, even for storyline visualization. In *Proceedings of the IEEE Conference on Visual Analytics Science and Technology*. 81–91.
[65] Han He and Jinho D. Choi. 2021. The stem cell hypothesis: Dilemma behind multi-task learning with transformer encoders. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. 5555–5577.
[66] Longhua Qian, Guodong Zhou, Fang Kong, Qiaoming Zhu, and Peide Qian. 2008. Exploiting constituent dependencies for tree kernel-based semantic relation extraction. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling'08)*. 697–704.
[67] Qi Li and Heng Ji. 2014. Incremental joint extraction of entity mentions and relations. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 402–412.
[68] Irina Kostitsyna, Martin Nöllenburg, Valentin Polishchuk, André Schulz, and Darren Strash. 2015. On minimizing crossings in storyline visualizations. In *International Symposium on Graph Drawing*. Springer, 192–198.
[69] Thomas C. van Dijk, Fabian Lipp, Peter Markfelder, and Alexander Wolff. 2017. Computing storyline visualizations with few block crossings. In *Proceedings of the International Symposium on Graph Drawing and Network Visualization*. 365–378.
[70] Emilio Di Giacomo, Walter Didimo, Giuseppe Liotta, Fabrizio Montecchiani, and Alessandra Tappini. 2020. Storyline visualizations with ubiquitous actors. In *International Symposium on Graph Drawing and Network Visualization*. Springer, 324–332.
[71] LLC Gurobi Optimization. 2022. *Gurobi*. https://www.gurobi.com/
[72] Ali Yadollahi, Ameneh Gholipour Shahraki, and Osmar R. Zaiane. 2017. Current state of text sentiment analysis from opinion to emotion mining. *ACM Computing Surveys (CSUR)* 50, 2 (2017), 1–33.
[73] Kurt Koffka. 2013. *Principles of Gestalt Psychology*. Routledge.
[74] Roy R. Behrens. 1998. Art, design and gestalt theory. *Leonardo* 31, 4 (1998), 299–303.
[75] Matthew Brehmer and Tamara Munzner. 2013. A multi-level typology of abstract visualization tasks. *IEEE Transactions on Visualization and Computer Graphics* 19, 12 (2013), 2376–2385.
[76] Cynthia Brewer. 2022. *ColorBrewer*. https://colorbrewer2.org/
[77] Martin Greilich, Michael Burch, and Stephan Diehl. 2009. Visualizing the evolution of compound digraphs with TimeArcTrees. In *Computer Graphics Forum*, Vol. 28. Wiley Online Library, 975–982.
[78] Fabian Beck, Michael Burch, Stephan Diehl, and Daniel Weiskopf. 2017. A taxonomy and survey of dynamic graph visualization. In *Computer Graphics Forum*, Vol. 36. Wiley Online Library, 133–159.

Received 13 December 2022; revised 18 October 2023; accepted 4 November 2023