




ThreadStates: State-based Visual Analysis of Disease Progression

Qianwen Wang , Tali Mazor, Theresa A Harbig , Ethan Cerami, Nils Gehlenborg 

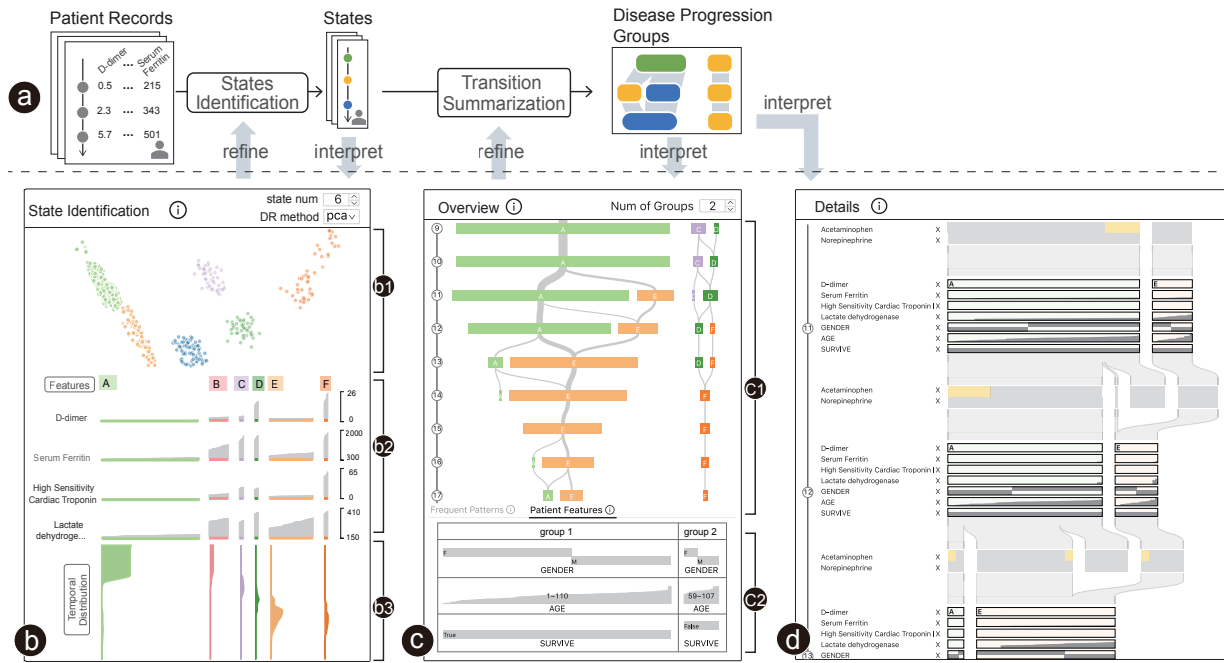


Fig. 1. *ThreadStates* supports an interactive disease progression analysis through two phases: state identification and transition summarization (a). For state identification, users can interpret and refine states through a novel matrix+glyph visualization and a scatter plot (b). For transition visualization, *ThreadStates* provides both an overview (c) and a detail view (d). State transition patterns are summarized and visualized using Sankey-based visualizations to reveal disease progression patterns.

Abstract—A growing number of longitudinal cohort studies are generating data with extensive patient observations across multiple timepoints. Such data offers promising opportunities to better understand the progression of diseases. However, these observations are usually treated as general events in existing visual analysis tools. As a result, their capabilities in modeling disease progression are not fully utilized. To fill this gap, we designed and implemented *ThreadStates*, an interactive visual analytics tool for the exploration of longitudinal patient cohort data. The focus of *ThreadStates* is to identify the states of disease progression by learning from observation data in a human-in-the-loop manner. We propose a novel *Glyph Matrix* design and combine it with a scatter plot to enable seamless identification, observation, and refinement of states. The disease progression patterns are then revealed in terms of state transitions using Sankey-based visualizations. We employ sequence clustering techniques to find patient groups with distinctive progression patterns, and to reveal the association between disease progression and patient-level features. The design and development were driven by a requirement analysis and iteratively refined based on feedback from domain experts over the course of a 10-month design study. Case studies and expert interviews demonstrate that *ThreadStates* can successively summarize disease states, reveal disease progression, and compare patient groups.

Index Terms—Disease Progression, State Identification, Sequence Visualization

1 INTRODUCTION

Longitudinal cohort studies play an important role in clinical research by contributing extensive patient observations cross multiple timepoints.

- Q. Wang and N. Gehlenborg are with Harvard University, USA.
E-mail: {qianwen_wang,nils}@hms.harvard.edu.
- T. Mazor and E. Cerami are with Dana-Farber Cancer Institute, USA.
E-mail: {tmazor,cerami}@jimmy.harvard.edu
- T. A. Harbig is with University of Tübingen, Germany.
E-mail:theresa-anisja.harbig@uni-tuebingen.de

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxx/TVCG.201x.xxxxxx

The data collected by these studies offer valuable information for researchers to track patient statuses and understand disease progression. For example, a number of studies [10, 18] aim to understand the development of cancer by recording and comparing patients' genomic mutations at multiple timepoints. Based on these observations, researchers are able to associate the gain of certain genomic mutations to the time of cancer relapse.

However, exploring and analyzing extensive datasets of a large patient cohort is a non-trivial task. First, cohort studies typically collect a wide variety of observations (e.g., antibody level, blood glucose, white blood cell) so that they can provide comprehensive information and do not overlook important data. The large number of observations, the complex correlation between observations, and the existence of irrelevant observations make it difficult to effectively extract meaningful patterns. Second, a disease can exhibit different progression patterns within a target patient cohort, varying from patient to patient and from

treatment to treatment. It is challenging to disentangle, identify, and present these patterns. Meanwhile, the identified disease progression patterns need to be associated with other variables (e.g., patients' genetic profiles, received treatments, health outcomes) in order to derive clinically meaningful insights. How to effectively reveal the association is crucial for the analysis of disease progression.

Many visual analytics methods have been proposed to facilitate the interpretation of health records in longitudinal cohort studies by integrating interactive visualizations with data mining algorithms. One main research direction is to formalize this problem as the visual analytics of temporal event sequences [6, 8, 12, 38, 39]. Guo et al. [13] reviewed and categorized the state-of-the-art methods in visual analytics of event sequences. Those studies proposed novel visual designs and interaction techniques and integrated them with data mining algorithms to address challenges in analyzing longitudinal clinical records, including the large number of sequences [39], the high dimensionality of events [8], and the large redundant patterns [6].

However, prior studies usually treated patient observations as general events and underutilized their power in modeling patient status. For example, DecisionFlow [8] visualizes lab tests (i.e., observations), diagnoses, and medications in health records as thousands of unique event types. While DecisionFlow enables effective aggregation and exploration of these events, it mixes observations with other events and provides little support in modeling patient status. Recently, DPVis [21] proposed visualization methods to analyze observations in clinical studies and track the progression of diseases. DPVis introduces the concept of **states** to describe different patient statuses using the value distributions of a set of observations. A Hidden Markov Model (HMM) is employed in DPVis to both summarize and make inferences of disease states. Multiple visualizations assist users in examination of the complex hyperparameters of the HMM and interpretation of the identified states. While DPVis sheds light on state-based visual analytics of disease progression, it only allows state modifications through adjusting the hyperparameters of the HMM, which can be challenging for clinical researchers. Instead of using an end-to-end model as in DPVis, we employ a phase-wise method to provide users more control over the process of disease progression analysis.

Here we describe how we developed *ThreadStates*, a web-based tool that provides state-based visual analytics for the discovery of disease progression patterns, based on a design study that was conducted over 10 months. *ThreadStates* helps users to effectively interpret, interact with, and analyze disease progression patterns that are learned from the values of regular patient observations. We propose an interactive workflow to assist users in the analysis through two phases: a *state identification* phase and a *transition summarization* phase (Fig. 1(a)). In state identification, an unsupervised clustering method is used to learn disease progression states from patient timestamp features. Users interpret different states through a novel *Glyph Matrix* design (Fig. 1(b2)), as well as a 2D projection of the states in a scatter plot. In transition summarization, common state transition patterns are summarized and visualized using a Sankey-based visualization (Fig. 1(c)). Patients are further grouped based on transition patterns to reduce visual clutter in the Sankey-based visualization and to reveal potential associations between progression patterns and other variables. We evaluate *ThreadStates* through case studies and expert interviews on two public datasets.

The main contributions are:

- A user-centered design study focused on visual analysis of disease progression. We summarize data characterization and analysis tasks, and propose an interactive visual analytics method for disease progress analysis through two phases: *state identification* and *transition summarization*. We discuss and justify considerations and rationales behind the algorithm choices and visualization designs.
- *ThreadStates*, an interactive web-based tool for state-based visual analytics of disease progression. This tool integrates existing sequence mining and clustering techniques with a set of novel visualization designs.
- Case studies and expert feedback that demonstrate the utility of *ThreadStates* and generate insights about disease progression.

2 RELATED WORK

The visual analysis of clinical sequence data falls into the general category of event sequence analysis. In this section, we discuss studies that are most relevant to *ThreadStates* and refer the readers to Guo's survey [13] for a comprehensive review.

To visualize event sequences, one straightforward solution is to place events along a time axis. This method has been employed in a number of studies, such as Lifelines [31], CloudLines [20] and ID-MVis [41]. However, this approach can lead to substantial cognitive load and hinder the identification of patterns as the number of sequences increases. To address this issue, EventFlow [27] and CoreFlow [23] extract branching structures from sequences and visualize them using tree-based visualization. Outflow [38], CareFlow [28], and DecisionFlow [8] condense sequences into transition graphs and visualize the sequences using Sankey-based visualizations. To reduce the visual clutter caused by the dense edges in such visualizations, MatrixFlow [29] and MatrixWave [42] propose novel matrix-based visualizations. Additionally, a number of interactions have been proposed to assist users in effectively querying sequences of interest and pruning of redundant sequences. (S)queries [40] and DecisionFlow [8] provide visual query interfaces. ID-MVis [41] allows users to stretch, annotate, and align sequences to facilitate pattern identification. EventPad [3] enables users to define rewrite rules using multivariate regular expressions for exploring multivariate event sequences.

To better facilitate pattern identification from sequences, an increasing number of visualization systems integrate interactive visualization with pattern mining algorithms. Frequency [30] and Peeksequence [22] employ sequential pattern mining algorithms and visualize mined patterns to assist users in understanding and analyzing event sequence data. Adapting the minimum description length principle, Sequence Synopsis [6] proposes a novel visual analytic framework that simultaneously clusters sequences and extracts cluster-level visual summaries. SeqCausal [17] extends this approach with causality analysis algorithms and enables users to explore, verify, and compare causalities in event sequences.

Stage analysis (i.e., finding frequently occurring subsequences) has been employed by several visual analytics studies to summarize sequences. For example, EventThread [12] uses tensor analysis to identify latent stages and visualizes stage progression patterns through interactive grouping. ET2 [11] aligns and segments sequences to support flexible stage intervals. StageMap [5] provides dynamic stage identification and supports exploring stages at different granularities.

These approaches target sequences that consist of a wide variety of event types, which allows them to support to a broad range of applications, such as the user click streams [42], vehicle fault analysis [6]. While these studies can be applied to clinical sequences with repeated observation data (e.g., glucose in type 1 diabetes), most of them did not consider and take full advantage of the unique characteristics of the repeated numerical measures. Several recent studies utilize the repeated observations in clinical sequences. For example, ID-MVis [41] targets the records of type 1 diabetes patients and the sequence data contains repeated observation data (e.g., blood glucose levels). However, ID-MVis focuses on folding, stretching, aligning sequences of individual patients, and provides little support for aggregation of sequences or investigation of patterns in a patient cohort. DPVis [21] mines and visualizes hidden states from repeated observation data in longitudinal clinical sequences. States are summarized in a *Feature Matrix* view, and state transition patterns are visualized in a *Pathway Waterfall* view. However, as discussed in the introduction, HMM is an end-to-end model and it is inconvenient for users to interact with the model and refine the state identifications.

ThreadStates is built upon these previous studies for the visual analysis of sequence data. We adopt a series of existing techniques into our system, such as sequential pattern mining, pattern query interactions, and sequence clustering. We also provide a set of user interactions to support searching, ranking, and pruning sequence patterns. *ThreadStates* employs a Sankey-based visualization and applies sequence clustering to reduce the edge crossing in each Sankey diagram. Similar to DPVis, *ThreadStates* conducts state-based analysis using the values

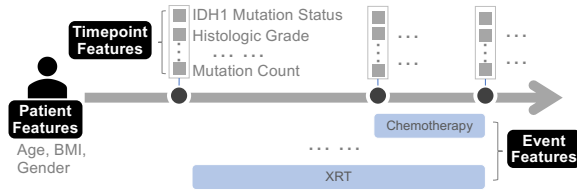


Fig. 2. An input data example from a low grade glioma dataset. Each sequence contains three types of features: patient features, timepoint features, and event features.

of observation data in clinical sequences. But unlike most previous studies, we target a common type of clinical sequences that contain regular repeated observation data. Unlike the end-to-end model that is employed in DPVis, we propose a two-phase workflow so that users can easily interact with and modify the identified states. We propose interactive state identification by learning from these repeated observation data and enable users to analyze disease progression in the context of state transitions. We also provide novel visualizations to facilitate the interpretation of state characteristics.

3 DESIGN METHODOLOGY

The design and development of *ThreadStates* is based on a user-centered approach that involved close collaboration between data visualization researchers and domain experts over 10 months.

3.1 Design Team and Design Process

ThreadStates is an extension of a previous study, *OncoThreads* [15], a visualization tool for longitudinal cancer molecular data. The requirements and an early prototype for *OncoThreads* were developed using the design sprint method [19] by eight researchers with backgrounds in cancer biology, biomedical informatics, and visualization over five consecutive days, for 6 hours each day.

This project was conducted by a subset of these researchers, including both visualization researchers and domain experts in biomedical informatics. Based on the initial design requirements and visualizations in *OncoThreads*, we further investigated how to conduct state-based visual analytics of disease progression in order to scale the analysis on high dimensional records and large number of patients. Our team met on a weekly basis to iterate visualization designs and tested the proposed visualizations on a tumor evolution study [18].

3.2 Data Characteristics

The input data of *ThreadStates* is a set of records of individual patients. As shown in Fig. 2, each patient is represented by a sequence of three types of features: timepoint features, event features, and patient features. **Timepoint features** are the observation data from individuals over a period of time, such as blood testing results per week and body temperature per day. This type of features describes patients' statuses at single timepoints and may change as the disease progresses, therefore playing a crucial role in the identification of disease states and modeling of disease progression. Timepoint features are usually high dimensional and may contain both numerical and ordinal values. Not all features might be available for all timepoints. **Event features** describe any event occurring between timepoints, such as drug treatments, surgery, or other clinical interventions. These events can influence the course of disease and play an important role in the analysis. Event features may have binary values, indicating whether an event has taken place or not, or more complex characteristics, such as dosage and treatment length for a particular drug. Unlike previous visualization tools that mainly consider point events (i.e., events that occur at specific timepoints), we support the investigation of interval events (i.e., events that last for a time duration). **Patient features** describe features that are related to patients. This type of features usually do not change with disease progression. Common patient features include age, gender, and disease outcome (e.g., alive or deceased). Previous studies have identified correlations between patient features and disease progression patterns [21, 35].

3.3 Domain Goals and Visual Analysis Tasks

The domain goals for *ThreadStates* are based on the high-level domain goals that Kwon et al. [21] reported for the study of disease progression through interactive visualization: summarize how features change to describe disease progression (**G1**); discover groups of patients who share changes in their features over time and are distinct from other groups (**G2**); identify correlations between progression trajectories and other features (**G3**). We validated these goals with domain experts and then translated them into the following visual analysis tasks.

T1. Characterize value distribution over multiple features at each state. States indicate patient statuses and are represented by the value distributions of multiple timepoint features. For example, cancer patients may exhibit different genomic mutations at the first cancer diagnosis and a later cancer relapse. Visualizing the timepoint features of each state enables users to interpret the identified states, compare different states, and refine states based on their analysis needs. As a result, users are able to understand disease progression through the evolution of these states and their timepoint feature values (**G1**).

T2. Summarize and compare the occurrence of each state. Users might be interested in the occurrence patterns of a state, such as when does a state normally occur and how frequently does this state occur in the target cohort. Since disease progression is depicted in terms of state transitions, summarizing the occurrence patterns of each state can offer users a more comprehensive understanding of states, therefore facilitating the interpretation of disease progressions (**G1**).

T3. Search, rank, and filter state transition patterns. Capturing important state transition patterns is crucial for the analysis of disease progression. It enables users to comprehend the evolution of diseases (**G1**) and find distinctive groups of patients with similar interested transition patterns (**G2**). Sequential pattern mining algorithms can generate long lists of patterns, many of which are redundant and do not contribute to the understanding of disease progression. Therefore, the tool should provide flexible support for users to quickly search, rank, and locate interested patterns.

T4. Group state transition patterns. Progression patterns of the same disease vary across patients and across treatments. Discovering and disentangling patterns from a target cohort allow researchers to obtain a comprehensive understanding of the disease. Grouping different transition patterns also provides an effective means to group patients and find patient subgroups with similar transition patterns (**G2**).

T5. Reveal the association between patient groups and other features. Usually, users not only want to summarize disease progression patterns but are also interested in revealing the association between progression patterns and other features (**G3**). These associations can generate valuable insights for a number of clinical tasks, such as early disease prediction and treatment selection. Since we group patients based on progression patterns, the association between progressions and other features can be revealed by analyzing the associations between patient groups and these features.

While identifying states is a key task (**T1**), the proposed analysis also applies to studies of diseases that have well-defined states. For these studies, this tool can help users examine the state distributions in the target patient cohort (**T2**), understand state transitions (**T3**, **T4**), and identify potential correlations between disease progressions, patient features, and received treatments (**T5**).

4 VISUAL ANALYSIS OF DISEASE PROGRESSION

The overall workflow of *ThreadStates* is described in Fig. 1. Contrary to an end-to-end method, we propose a two-phase workflow to give users more control over the disease progression analysis process. In the first phase, an unsupervised clustering method learns disease states from patient timepoint features. Users interpret different states through a set of visualizations and update the state identification results based on their analysis needs. In the second phase, transitions among the identified states are summarized and visualized. Patients are further grouped based on transition patterns to reduce visual clutter and to reveal associations between progression patterns and other features.

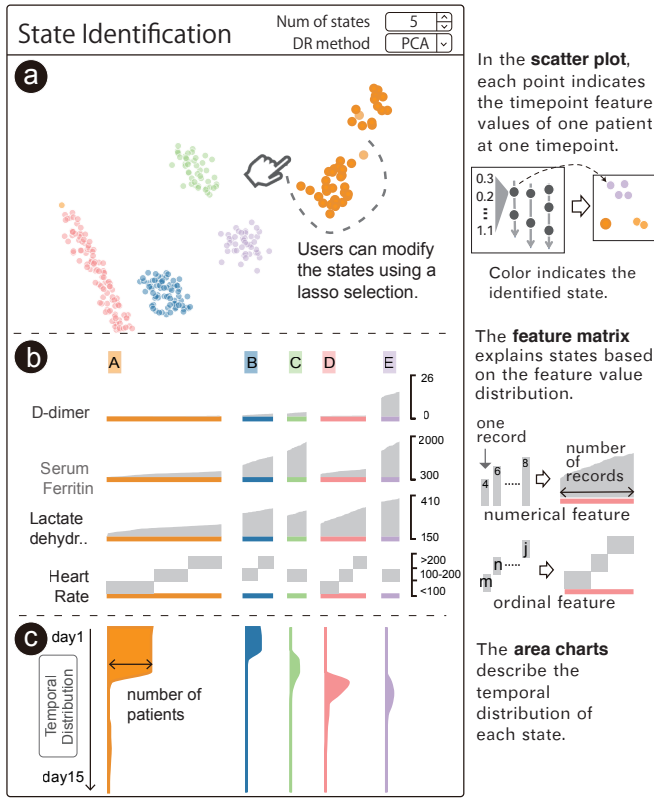


Fig. 3. The state identification view consists of three visual components: a scatter plot (a), a *Glyph Matrix* (b), and an area chart (c). (a): The scatter plot projects the high dimensional timepoint features into a 2D space. (b): The *Glyph Matrix* employs a novel glyph design and enables easy comparison across both states and features. (c): The area chart summarizes the temporal distribution of each state.

4.1 Phase 1: Identify, Interpret and Refine States

Algorithm Choice. In *ThreadStates*, we treat state identification as an unsupervised clustering task. Each input item is a high dimensional vector $O_{i,j} = (x_1, x_2, \dots, x_m)$. $O_{i,j}$ indicates the observation of patient i at timepoint j . (x_1, x_2, \dots, x_m) indicate the value of the m types of observations, such as body temperature, antibody level. The task is to find clusters with similar timepoint observations. For state-based visual analysis of disease progression, previous studies have employed Hidden Markov Models (HMM) to identify states and state transitions synchronously [21]. The synchronous identification of states and transitions enables efficient pattern mining but provides little support for user participation and state refinement. In this study, we use a hierarchical agglomerative clustering method [32] to provide users more flexibility in the process of state identification. For missing values, we replace

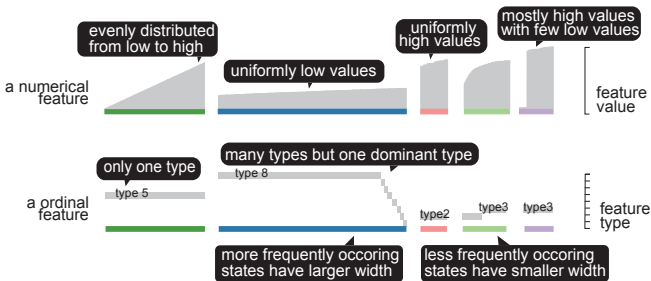


Fig. 4. The proposed glyph captures various feature distribution patterns (see the annotations in black boxes) to aid in the interpretation of the characteristics of different states.

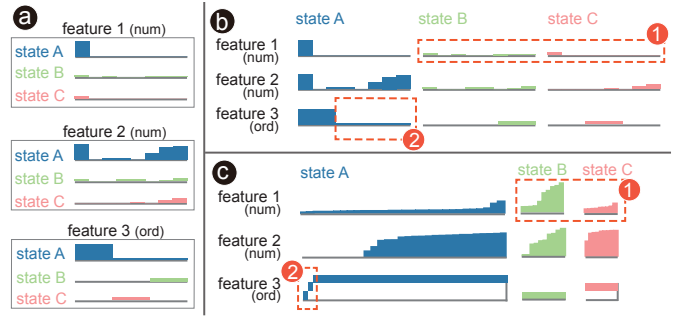


Fig. 5. Design iteration of the *Glyph Matrix*. The *Feature Cards* design (a) visualized and compared states at each feature. This design was later updated to a matrix-based design (b) to enable efficient summarizing of each single state. Since histograms in the matrix-based design can be ineffective in comparing the occurrence frequency and the value distribution (red boxes), we proposed a novel glyph design (c).

them with non-missing values at the nearest timepoint of the same patient, based on the assumption that clinicians are more likely to skip an observation that has no dramatic change.

The clustering results are influenced by the selected timepoint features. The selection of timepoint features will influence the state identification results, therefore affecting the transition summarization and the analysis conclusions. The integration of users' domain knowledge can help select appropriate features and identify clinically meaningful states. In our earlier work [15], we proposed a *Feature Manager* based on Lineup [9]. This *Feature Manager* calculates a set of variability scores to highlight features that are of potential clinical interest.

Visualization. It can be challenging for users to interpret identified states and refine the identification. Therefore, we provide a set of interactive visualizations that enable users to interpret the state identification from different aspects (Fig. 3). A **scatter plot** enables users to observe the state identification results in a 2D space. As shown in Fig. 3(a), each circle indicates one timepoint of one patient. Its position indicates the feature values of this timepoint, and its color indicates the identified state. We support three widely-used dimension reduction methods: PCA [37], t-SNE [34], and UMAP [24]. The features used in the dimension reduction are the same ones used in clustering. Users can freely choose one method that works the best for their dataset. We found that PCA generates satisfactory results in most cases, since its linear nature enables users to directly connect the projection results with the distribution of feature values. To help users better understand states from the original high dimensional space, we propose a novel *Glyph Matrix* that explains the characteristics of states based on their value distribution on different features (T1.). As shown in Fig. 3(b), the *Glyph Matrix* employs a small multiples design [33]. Each small chart depicts the value distribution of one state (column) on one feature (row). This *Glyph Matrix* design is similar to the matrix design proposed in DPVis. However, DPVis only provides statistic summaries (e.g., mean, standard deviation) for the feature distribution. The occurrence frequency (i.e., number of records) of each state is not presented to users in DPVis. This information can be important for users to better understand the occurrence of each state (T2.). Users may try to read the occurring frequency from the scatter plot, but the overlapping of points makes it hard to read the number of points accurately. Therefore, we propose a novel glyph design to depict the feature distribution at each cell. The width of the glyph indicates the number of records belonging to this state. For numerical features, the height of the glyph indicates the value number; for ordinal features, the y position of the rectangle indicates the value type. As shown in Fig. 4, this chart can depict a variety of distribution patterns for both numerical and ordinal features. Sometimes, the values in one cell are small and the height of the glyph is too low to observe. We provide a tooltip in which the glyph is scaled to enable effective observation of the value distribution. *ThreadStates* also provides **area charts** to present the temporal distribution of each state (T2.). As shown in Fig. 3(c), users can easily compare the temporal distributions of states to better understand their role in disease progression.

Design Iteration The *Glyph Matrix* design was iterated several times based on input from the domain experts on the author team. Alternative designs were proposed and tested during each design iteration. As shown in Fig. 5, we first proposed *Feature Cards* (a). For each feature, we visualized the value distribution of each state using a histogram (for numerical values) or a bar chart (for ordinal values). While the domain experts confirmed the usefulness of design (a), they commented that this design spread the characterization of a state spread over a number of *Feature Cards* and made it inconvenient to summarize one state. The domain experts expressed a preference to a matrix based design (b), which enables easy comparison across both states and features. However, in design (b), it is not optimal to compare states on one feature, i.e., comparing histograms horizontally. Also, histograms represent states with low occurrence frequencies using low bars, making it difficult to read the feature value distributions. For example, as shown in Fig. 5(1), it is hard to examine and compare state B and state C in design (b). The histograms may also lead to the overlooking of certain value distributions (e.g., Fig. 5(2)). Meanwhile, when using histograms and bar charts, it is ineffective to compare the occurrence frequency of states, which is important for the proposed analysis (T2). Users can only estimate the occurrence frequency of each state by summing up the height of all rectangles. To address this issue, we proposed a novel glyph for the *Glyph Matrix* (c). This glyph reflected a design trade-off between the ease of reading certain important information and the familiarity of the visualization. We further refined the design by removing glyph color to facilitate comparison of glyphs in the same row, i.e., comparing states across one feature.

4.2 Phase 2: Group and Compare Transition Patterns

After phase 1, patient i can be represented by a sequence of identified states $P_i = (s_1, s_2, \dots, s_k), s \in S$, where S is the space of all identified states and k is the number of timepoints. In phase 2, we aim to identify 1) frequent state transition patterns and 2) patient groups with similar transition patterns. Note that the current version of *ThreadStates* focuses on the analysis of sequences and does not consider the interval between timepoints. Event features are considered based on their relative positions to these timepoints rather than their actual timestamps.

Algorithm Choice. Previous studies have proposed a wide range of novel algorithms to tackle various challenges in the visual analytics of sequence data. Instead of proposing additional algorithms, we focus on experimenting and selecting a suitable algorithm from existing ones based on the analysis needs in disease progression.

Stage summary algorithms: Stage analysis can provide a concise presentation of sequences by identifying stages through frequently occurring subsequences. Previous studies have contributed algorithms and visualizations for stage summary [5, 11, 12]. However, this type of

algorithm divides each individual sequence into segments, yet interesting transition patterns often overlap with each other. Moreover, cohort studies usually already include stage-related information in timepoints, e.g., diagnosis, remission, and relapse in cancer studies.

MDL-based algorithms: The Minimum Description Length (MDL) principle is a general criterion for model selection in inductive inference. MDL-based methods have successfully been applied in earlier studies for the visual analysis of event sequences to achieve a good balance between the simplicity of the visual representation and its information content [6]. Even though MDL-based methods can effectively summarize coarse-level patterns of the sequences, they fail to reveal local state transition patterns, which were identified as crucial by the domain experts on our team.

SPM-based algorithms: Sequential Pattern Mining (SPM) applies frequency-based mining algorithms to identify patterns in sequences. It can identify both short and long patterns, regardless of whether they are overlapping with each other. Since SPM can address the issues we identified with stage summary and MDL-based algorithms, we employed it for the discovery of transition patterns in *ThreadStates*.

We first experimented with PrefixSpan [14], a widely-used sequential pattern mining method. PrefixSpan introduces redundant patterns, which can hinder effective visual analysis. Therefore, to reduce redundant patterns and improve the analysis efficiency, we mainly considered transition patterns in consecutive timepoints. This decision is based on the discussion with the domain experts on our team. We ultimately chose to employ N-grams, an information retrieval approach for representing subsequences of words from document collections.

N-gram based methods have exhibited great performance in previous event sequence analysis studies [2, 36]. Based on those previous studies, we implemented the algorithm in *ThreadStates* as below. Each patient P_i can be represented by a set of N-grams, i.e., N consecutive elements in the state sequence (s_1, s_2, \dots, s_k) . We denote the N-gram set of patient P_i as $T_N(P_i)$, which can be represented as $T_N(P_i) = \{(s_j, s_{j+1}, \dots, s_{j+N-1}) | j \in [1, k+1-N]\}$, where k is the number of timepoints, N is the number of consecutive elements. The distance between patient i and patient j is calculated as the Jaccard distance between the two N-gram sets: $dist(P_i, P_j) = T_N(P_i) \cap T_N(P_j) / T_N(P_i) \cup T_N(P_j)$. Patients are then grouped through hierarchical agglomerative clustering [32] based on their pairwise distances. In other words, patients with similar state transition patterns will be grouped together (T4).

The setting of N depends on the properties of the dataset. We chose N as [2, 3] for the testing dataset [18]. We found that increasing N does not give many benefits after a certain threshold. First, large N captures longer transitions that are less likely to repeat as a pattern. Second, the number of patterns increases exponentially with N and significantly increases the associated computational costs.

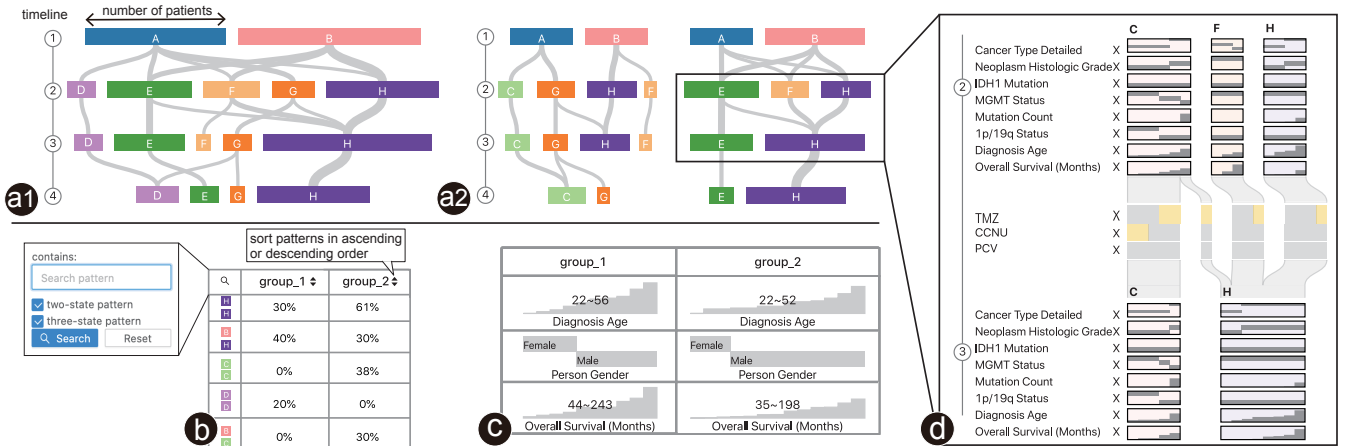


Fig. 6. The disease progression patterns are expressed in terms of state transitions. Transitions are visualized using Sankey-based visualization and sequence clustering is applied to reduce the edge crossing (a). A pattern table (b) enables users to search, sort, filter the frequent patterns of each group. A patient table (c) summarizes the patient features of each group to reveal the correlation between state transition patterns and patient features. A detail view shows the distribution of feature values and provides more details about selected state transitions (d).

Visualization. Visual analysis of event sequence data can be categorized into five major groups: chart-based, timeline-based, hierarchy-based, Sankey-based, matrix-based [13]. As with our previous work [15], we chose Sankey-based visualization because it is intuitive, widely-used, and can effectively aggregate the transition between timepoints. As shown in Fig. 6(a), each rectangle represents patients with the same state at a specific timepoint: its color indicates the state; its width indicates the number of patients; position in the vertical direction indicates the timepoint. Flows between the rectangles reveal patients’ states transition between timepoints. Since the state identification results will significantly influence the transition summary, *ThreadStates* allows users to refine the state identification based on the results of transition summarization and iterate between the two phases.

The visual clutter that arises from a growing number of transition paths is a widely-recognized shortcoming of Sankey-based visualizations. This problem can be mitigated by our proposed sequence grouping. As shown in Fig. 6(a2), grouping patients dramatically reduces the visual clutter and reveals the transition patterns. Users can change the number of patient groups, merge or split the identified states to best suit their analysis needs. Users may want to analyze detailed information about a state transition, such as how a certain attribute changes between timepoints. Therefore, *ThreadStates* complements the Sankey overview with a detailed view, as shown in Fig. 6(d). In the detailed view, rectangles in the overview are expanded into multiple rows where each row represents patients’ value distribution over one feature using the same glyph as in *Glyph Matrix*. Event features are visualized as rows in the flows based on their relative positions to the timepoints.

Apart from the Sankey-based visualization, *ThreadStates* provides two tables to assist the analysis of different patient groups. The first table (Fig. 6(b)) enables users to search, sort, filter the frequent patterns of each patient group to better understand the transition patterns (T3.). The second table (Fig. 6(c)) summarizes the patient-features of each group to reveal the correlation between state transition patterns and patient features (T5.). In each table cell, the patient feature visualization is represented using the same glyph as the one used in the *Glyph Matrix* (Fig. 3(b)) to reduce the learning burden.

Design Iteration. The state transition view was iterated during the design study based on our weekly team discussions. During this process, we identified a weakness of Sankey-based visualizations that is rarely discussed in previous studies: Sankey-based visualizations can give users a false sense of transitions between multiple timepoints even though it does effectively summarize the transition between two consecutive timepoints. The visual clutter caused by dense edge crossing in Sankey-based visualizations has been widely studied, but this ambiguity can exist without dense edges and has not been discussed in depth. As shown in Fig. 7, users may identify a stateB-stateC-stateA (■ ■ ■) transition even though this transition does not actually happen. Sometimes, users also overlook some existent transitions, such as the stateA-stateC-stateB (■ ■ ■) transition, even though they do exist. This visual ambiguity can confuse users and have a negative impact on the analysis. We propose three strategies to tackle this issue from different angles. First, we show the frequency of each pattern in the pattern table (a). Users can query and validate the existence of one specific transition pattern. Second, we enable users to interactively change the number of patient groups and divide a single group into multiple subgroups (b). Third, we enable users to select nodes or flows of interest to highlight their subsequent transitions (c). Meanwhile, considering Sankey-based visualizations are widely used in the visual analytics of event sequences and a wide range of other applications, we believe this visual ambiguity is worth to be further investigated.

5 IMPLEMENTATION

ThreadStates is a web-based interactive visualization tool implemented in JavaScript using *React* [16], *mobx* [26], and *D3* [1] for the application structure, state management, and visualization, respectively. An interactive onboarding guide is provided using *Intro.js* [25]. The source code and an online demo are available at <http://threadstates.gehlenborglab.org> under the MIT license.

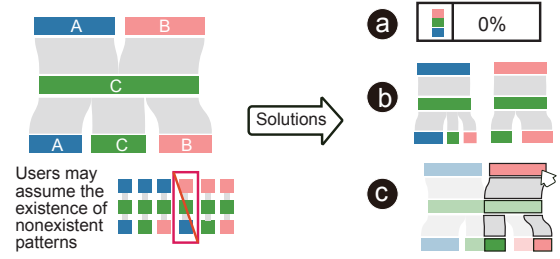


Fig. 7. Sankey-based visualizations may lead users to assume the existence of nonexistent patterns. We proposed three solutions to mitigate this problem: directly list patterns in the pattern-table (a); interactively group patients based on transition patterns (b); highlight selected individuals in the Sankey-based visualization (c).

ThreadStates is built as an extension of *OncoThreads* [15], our previous study on visualization of large scale longitudinal cancer molecular data. *ThreadStates* reuses several features from *OncoThreads*, including the *Feature Manager* and *Feature Explorer*, rich interactions in the Sankey-based visualizations, data processing for various patient records and observations, and supports for provenance tracking and visualization export. *ThreadStates* extends *OncoThreads* by introducing interactive visualizations for state identification and sequence clustering, rendering the analysis on high dimensional observations and large numbers of patients more scalable. *ThreadStates* allows users to upload their datasets from local computers or import data through the cBioPortal [4, 7] Web API. Note that *ThreadStates* provides comprehensive support for cohort study analysis. In this paper, we focus on the features that are most relevant to the state-based analysis.

6 CASE STUDIES

We evaluated *ThreadStates* by conducting case studies with domain experts on two public datasets: a synthetic COVID-19 dataset [35] and a real cytogenetically normal acute myeloid leukemia (CN-AML) dataset [10]. The first dataset has relatively long sequences with dense timepoint features, while the second dataset has relatively short sequences with sparse timepoint features. To demonstrate the effectiveness and generality of *ThreadStates*, none of the case study datasets were used during the design and development process. For the first dataset, we collaborated with a practicing intensive care physician who conducts clinical informatics research. This domain expert, referred to as E1, is not a collaborator in this project nor a co-author. E1 was assisted during the case study and asked questions freely. They then received the interview immediately after the case study. For the second dataset, we collaborated with a cancer biologist, referred to as E2, who conducts research on cancer genomics. E2 is a co-author of this paper and participated in the design process. E2 first performed the analysis described in the use case independently. They then reported the findings and received the interview.

6.1 Case Study 1: The disease progression of COVID-19

In this case study, we analyzed the disease progression of COVID-19 using a synthetic public dataset contributed by Walonoski et al. [35]. We analyzed the medical records of 100 simulated patients from that dataset. For these patients, a daily measurement for a set of indicators (such as heart rate, white blood cell count, etc.) had been generated using a model-based approach based on real data to protect patient privacy. Given the synthetic nature of the data, we did not aim to generate testable hypotheses but demonstrate the effectiveness of *ThreadStates*.

We first loaded several interested timepoint features, *D-dimer*, *Serum Ferritin*, *High Sensitivity Cardiac Troponin I*, and *Lactate Dehydrogenase*, and applied PCA for dimension reduction. As shown in Fig. 1(b1), there were six clusters in the scatter plot with clear boundaries. We set the number of states to six and *ThreadStates* automatically summarized each state. As shown in Fig. 1(b2) and (b3), the identified states demonstrated clear differences from each other: on the feature value distributions, on the temporal distributions, and on the occurring

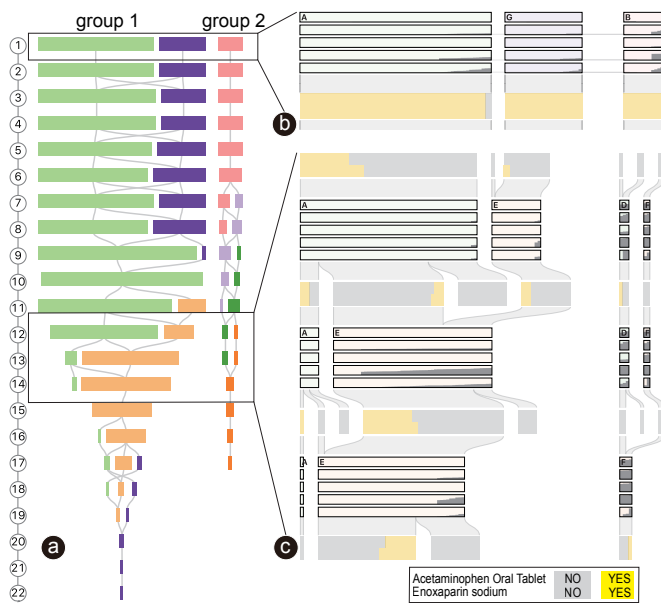


Fig. 8. In the COVID-19 dataset [35], patients are grouped into two groups with distinct state transition patterns. Users can analyze the state transitions from both an overview (a) and a detail view (b,c).

frequency. In terms of value distribution, we observed that all charts in the stateA column had relatively low heights, indicating low values of all four features. On the contrary, charts in the stateB column had relatively high heights, indicating high values of the four features. In terms of temporal distribution, stateA and stateB usually occurred at the early stage of the disease, while stateF usually occurred at the late stage. In terms of occurring frequency, the glyphs of stateA were much wider than other glyphs, indicating a much higher occurring frequency in the patient cohort. Since the cluster of stateA consisted of two clusters that were slightly overlapping, we manually split stateA into two states using a lasso selection in the scatter plot.

Based on the identified states, the state transitions can be grouped into two distinctive groups, as shown in Fig. 8(a). Patient group1 started at either stateA or stateG. Patients then switched between these two states (day 1 to day 8). Most patients later transferred to stateA and then gradually changed to stateE. At the end, some patients came back again the stateG. Contrary to group1, the state transitions in group2 was simple and did not switch back and forth. Most patients follow the transition from stateB to stateC to stateD and then to stateF.

Comparing patient features of the two groups, we can identify associations between transition patterns and patient features. As shown in the patient table (Fig. 1(c2)), all the patients in group1 survived while none of the patients in group2 did. The two groups of patients also exhibited different distributions of age and gender. For group2 (all patients deceased), only a small proportion of the patients were females, and all patients were older than 59.

We then explored how other timepoint features could influence the state identification and the disease progression patterns. We opened the feature explorer and ranked all the features based on the rate of change (Fig. 9(a)). We added the top four features: *Heart Rate*, *Respiratory Rate*, *Body Temperature*, *C-Reactive Protein*. However, after adding the four features, the clusters in the scatter plot disappeared and all points were mixed together (Fig. 9(b)). This indicated that the added new features, even significantly changed in the patient cohort, did not contribute to the state identification. In the *Glyph Matrix* (Fig. 9(c)), we observed that, at the rows of *Heart Rate*, *Respiratory Rate*, and *Body Temperature*, glyphs were right-angled triangles with the same heights. This indicated that all states had similar value distributions in these features. More specifically, the feature values were evenly

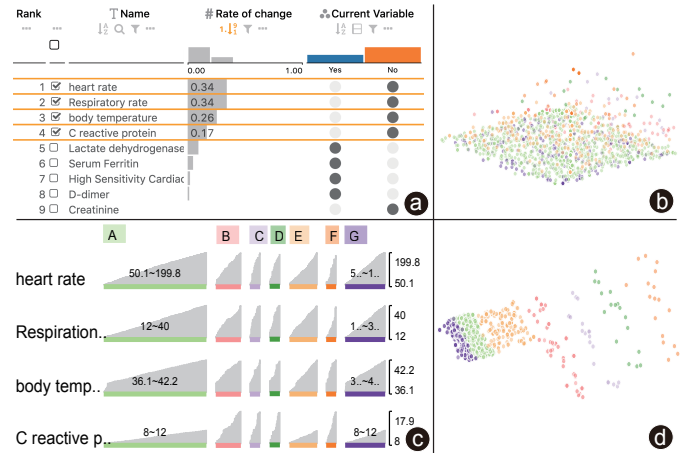


Fig. 9. (a): All features were ranked based on the rate of change and we selected the top four features. (b): Points were mixed together after adding the four new features. (c): The *Glyph Matrix* showed the value distribution of each state (column) on the four new features (row). At the rows of Heart Rate, Respiration Rate, and Body Temperature, glyphs were right-angle triangles with the same heights. This indicated the corresponding feature had the same even distribution for each state. (d): After removing Heart Rate, Respiration Rate, and Body Temperature, the points formed clusters.

distributed from low to high. For *C-Reactive Protein*, while all glyphs are right-angled triangles, the glyphs at stateA, stateE, and stateG were much shorter than glyphs at other states. This indicated that states had different value ranges at *C-Reactive Protein*. We therefore kept *C-Reactive Protein* and removed the other three features. The scatter plot changed to Fig. 9(d).

We then added event features (medication in this case) and analyzed their role in disease progression. Added two medications of interest, *Enoxaparin Sodium* and *Acetaminophen*. Each medication usually has several variants. For example, *Acetaminophen* has *500 mg oral tablet* and *325 mg oral tablet* in the dataset. To facilitate the analysis, we combined variants of the same medication using a logic “OR” operator. In the detailed view, we observed that the two medications were usually applied together in this dataset. The medications were given to more patients at the beginning of the disease (day1, Fig. 8(b)) than in the later stage of the disease (day12 - day14, Fig. 8(c)). Fig. 8(c) also showed that the two medications were often applied when the values of timepoint features were low, i.e., the glyph in each row had a low height.

In summary, *ThreadStates* was able to identify states based on multiple feature distributions, summarize common transition patterns, and form distinctive patient groups. Therefore, *ThreadStates* enabled a more profound and comprehensive analysis compared with the box-plot-based analysis conducted by Walonoski et al. [35], which only analyzed the change of individual features and failed to reveal how these feature values are associated with each other or how the disease progression patterns are influenced by patient features.

6.2 Case Study 2: The disease progression of CN-AML

In this case study, we explored the evolution of CN-AML using the dataset published by Greif et al. [10]. This dataset provides the somatic genomic alterations of 50 patients at three critical timepoints of CN-AML progression (i.e., diagnosis, remission, relapse). In this case, we aimed to evaluate *ThreadStates* by reproducing the findings in [10] and coming up with new insights and corroborating evidence.

Initially, we loaded gene mutations that frequently occurred in this patient cohort. As shown in Fig. 10, after applying PCA for dimensionality reduction, points in the scatter plot formed four clusters, indicating four identified states. The occurring frequency decreased in order of stateC>stateA>stateB>stateD, as revealed by the width of corresponding glyphs. Based on the *Glyph Matrix*, we found that the

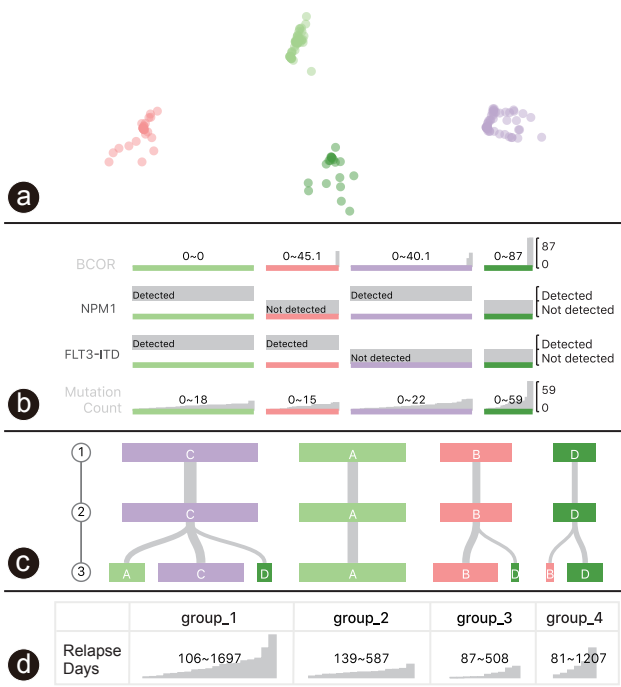


Fig. 10. (a): Points in the scatter plot formed four clusters. (b): From the *Glyph Matrix*, the main difference between the four states were whether *NPM1* and *FLT3-ITD* were detected. (c): *ThreadStates* grouped patients into four groups based on the identified states. (d): In patient table, we compared the four patient groups in terms of *Relapse Days*.

four states were formed based on whether *FLT3-ITD* mutation and *NPM1* mutation was detected. In other words, the mutation status of *FLT3-ITD* and *NPM1* rarely changed between diagnosis and relapse. Different patient groups demonstrated distinctive distributions of patient features. Compared with other groups, the cell glyphs in group2 and group3 had low heights, suggesting that patients in the two groups may have a shorter time to relapse (Fig. 10(d)).

Since the states that we identified initially characterized the patients rather than disease progression, we modified features used for state identification to find states that can better depict the progression of CN-AML. We removed *FLT3-ITD* and *NPM1*. That resulted in the points in the scatter plot being mixed together without discernible clusters present. We changed the dimension reduction method to UMAP, which generated better clustering results but still no meaningful states. We hypothesized that this was caused by the existence of many irrelevant features. Therefore, we removed other features and focused on mutations of *DNMT3A*, which was the most frequent mutation besides *FLT3-ITD* and *NPM1* in this cohort. We added three features related to *DNMT3A*: *Mutation Type* (none/wild type, truncation, in-frame, promoter, missense, other), *Protein Change* (indicating changed amino acids in the protein sequence), and variant allele frequency (*VAF*). In the *Feature Explorer*, we discretized *VAF* into three bins (i.e., $< 5\%$, $5 - 25\%$, $> 25\%$) based on its value distribution.

With the modified features, we observed three states. As shown in Fig. 11(a), stateA was characterized by the absence of *DNMT3A* mutation, as shown by *Protein Change* and *Protein Type* dominated by “wild type” and $VAF < 5\%$. Both stateB and stateC were characterized by the presence of *DNMT3A* mutation. StateC had relatively low *VAF* ($< 5\%$) while stateB has mutations with high *VAF* ($> 25\%$). StateB showed a variety of *Protein Change* but the most dominant is R882H (the corresponding rectangle had the longest width). The occurring frequency of stateC was visibly less than stateA and stateB.

We then used the identified states to group patients. We first grouped patients into two groups: the first group started with stateA and the second group started with stateB. In the patient table, we noticed that,

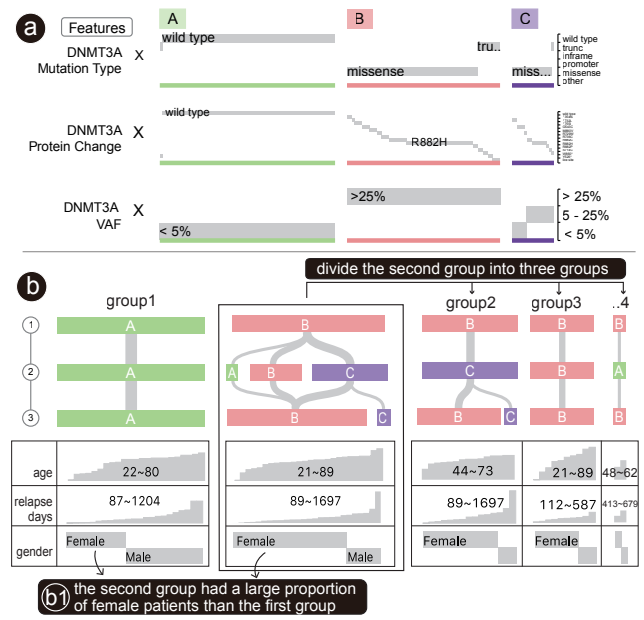


Fig. 11. (a): In the CN-AML cohort [10], we can identify three states based on features related to *DNMT3A*. (b): We group patients based on state transition patterns and compare patient features across groups.

in the column of the second group, the rectangle indicating Female was much wider than the rectangle indicating Male, as shown in Fig. 11(b1). In other words, the presence of *DNMT3A* (i.e., $VAF > 5\%$) was more likely to occur in female patients at diagnosis in this cohort. In terms of Age and Relapse Days, there was no visible difference between the two groups. To analyze the association between patient groups and Relapse Days, we further divided the second group into three groups (i.e., group2, group3, group4) and compared the four groups. All patients in group1 stayed in stateA and did not change. One main difference between stateA and the other states is that stateA had *DNMT3A VAF* $< 5\%$, which was considered as “not detected” by Greif et al. [10]. Without the presence of a *DNMT3A* mutation, the number of days to relapse of patients in group1 varied from very low to very high, as indicated by the slope of the corresponding glyph. Apart from group1, all other groups started in stateB, which was the only state that had *DNMT3A VAF* $> 25\%$. If the patients remained in stateB (i.e., group3), they had a shorter relapse time (i.e., the corresponding cell glyph had a low height). If the patients changed into stateC at the remission timepoint, even though most of them came back to stateB later, the relapse situation was much improved, and a proportion of the patients had higher values for Relapse Days, which was indicated by a high peak in the corresponding cell glyph. The improvement of relapse also happened in group4, where patients changed to stateA at the remission timepoint. The glyph of group4 did not include the low values that existed in the glyph of group3, indicating that group4 did not have patients with very short relapse. It is important to note, however, that while group4 represented a unique state transition pattern, it was a small group (the glyph had a small width). Whether the observation made on the small patient count can lead to biologically meaningful conclusions requires further investigation.

In summary, changes of *FLT3-ITD* and *NPM1* mutations were less likely to be observed during the disease progression. For patients with *DNMT3A* mutations at diagnosis, higher *VAF* at remission was suggestive of a shorter time to relapse, in line with the results discussed by Greif et al. in [10]. We also found that the presence of *DNMT3A* mutations at diagnosis was more common in females. This was a new finding that was not mentioned by Greif et al. in [10].

6.3 Expert Interview

After the case studies, we conducted semi-structured interviews with two domain experts. Here we report our observations from the case studies and our discussions with the domain experts in the interview. Note that we did not use a User Experience Questionnaire (UEQ) due to our limited number of participants.

Learnability and Usability. Both experts stated that they could quickly learn and understand *ThreadStates*. They both agreed that they were able to read the novel cell glyphs and interpret the meaning of the identified states. Both experts expressed that *ThreadStates* contributed an effective way to explore and analyze cohort study data. E2 commented the *State Identification* view was the most useful for their analysis. They stated that “*this view helped me identify and interpret states by considering several features simultaneously. It enabled analysis that is beyond my brain, which can only focus on one feature at one time*”. E2 also suggested adding statistic summaries (e.g., mean, median) in the cell glyphs to better assist the analysis. E1 liked the ability of *ThreadStates* to separate patients into different groups. They agreed that the comparison between patient groups can effectively derive insights. They also mentioned that “*this is exactly what we do in clinical studies. We form different patient groups and conduct comparisons*”. E1 expressed high interest in collaborating and applying *ThreadStates* to their own datasets.

Meanwhile, both experts also stated that it was not easy to choose suitable values for the number of states and the number of patient groups. E2 stated that “*I am not sure whether I set the right number, especially when exploring a new dataset*”. Even though interacting with the data and trying different values could mitigate this issue, they would like clearer guidance, such as providing a default value based on statistics of the dataset.

Analysis Workflow. Both experts thought that the two-phase workflow “*makes sense*” and “*generates reasonable results*.” While they trusted the automatic results, such as the frequent patterns and the identified patient groups, they also wished to have more control over the results. For example, E1 expressed the desire to group patients in a more customized way. In spite of some differences between *ThreadStates* and their original workflows, experts can quickly understand the workflow in *ThreadStates* and use *ThreadStates* to conduct analysis. For example, E2 usually start their analyses by observing individual patient records using tables or simple charts. They said that they could quickly shift their mindsets and start interpreting the dataset in the context of disease states.

We also observed that in these case studies experts adopted the workflow according to their own analysis contexts. E1, who conducts research on clinical data using informatics and statistical approaches, was more interested in using the *Detail View*, where they could compare different patient groups and analyzed how medications were associated with disease progression. E2, who conducts research in cancer genomics, was more interested in comparing and interpreting different states, which allowed them to identify the association between different gene mutations.

7 DISCUSSION

Limitations. While the case studies have demonstrated the effectiveness of *ThreadStates*, the tool still has several limitations. First, *ThreadStates* automatically aligns discrete timepoints based on a reasonable assumption that the repeated measure are regular in some cohort studies. The effectiveness of *ThreadStates* on irregular timepoint observations has not been tested. Second, we use an N-gram-based algorithm to strike a balance between effectiveness and accuracy. This algorithm emphasizes patterns between consecutive timepoints. However, transition patterns between non-consecutive timepoints can be important in some cases. How to effectively capture diverse patterns and remove redundant patterns in sequential pattern mining is still an open question in data mining. When using N-grams, the proper value for N needs to be carefully chosen based on the target dataset. Third, we conducted dimension reduction for the scatter plot visualization. Most existing dimension reduction methods are designed for numerical data. Therefore, in *ThreadStates*, we convert ordinal features to numerical features following the practices in previous studies. Such a method has

achieved satisfactory results at our cases. The dimension reduction of mixed type data requires further advances in algorithm designs. Lastly, the clustering results (i.e., the identified states) are influenced by the selected timepoint features and the defined number of states. Any selections made in the state identification phase will be propagated to the transition summarization and affect the conclusions. *ThreadStates* enables users to participate in the two-phase workflow and use their domain knowledge to facilitate the analysis. However, as with most exploratory analyses, it is still an open question how to better validate the selected parameters and the obtained insights.

Scalability. By summarizing patients records as states and aggregating sequences into transition groups, *ThreadStates* can scale well with the number of patients and the number of record types. However, the effectiveness of *ThreadStates* is limited by the number of states and the number of timepoints. We use a categorical color scheme to encode different states. This visual encoding is limited to analysis scenarios with a few number of states. The *Glyph Matrix* may become less effective with an increasing number of states (matrix columns). Meanwhile, a large number of states will make it hard to summarize meaningful transition patterns. The complexity of the Sankey-based visualization increases with the number of timepoints, making it hard to extract insight. In the future, we plan to further improve the scalability of *ThreadStates*. To improve scalability with the number of states, we plan to support a hierarchical state summarization mechanism. To improve scalability with the number of timepoints, we will allow users to segment and fold sequences at different levels of granularity.

Generality. While *ThreadStates* is designed and developed for the analysis of disease progression, it can easily be generalized to other high dimensional temporal sequences, such as air quality monitoring data. The *Glyph Matrix* design can be adapted to other applications that need to summarize and compare the high dimensional feature distributions of multiple groups. The novel glyph we proposed can be used as an alternative design to traditional histograms, especially when there is a need to compare the sum of items of two distributions.

8 CONCLUSION

In this paper, we described a design study on the visual analytics of disease progression with domain experts. Based on this design study, we propose *ThreadStates*. This interactive visualization tool enables users to conduct state-based visual analytics of disease progression in a two-phase workflow: state identification and transition summarization. We characterize the input data, summarize the analysis tasks, and discuss the design rationales behind *ThreadStates*. Apart from using several common visualizations such as scatter plots and area charts, we also introduce a novel *Glyph Matrix* visualization to assist users in the interpretation and comparison of different states. We anticipate that this *Glyph Matrix* design can be generalized to applications beyond disease progression analysis. This visualization design can effectively summarize and compare the value distribution (for both numerical and ordinal data) of high dimension features across multiple groups. The proposed glyph can be used as an alternative design to traditional histograms, especially when users need to compare the sum of items. As illustrated in our case studies, states that are identified from timepoint features can capture the progression of diseases and reveal the association between disease progression and other variables. We expect that our findings will inform future visual analytics studies on electronic health records data that go beyond categorical event types and consider the observed measures that are repeated across timepoints.

ACKNOWLEDGMENTS

This project is supported by the grant “Drug Discovery & Translational Research Program” from Novartis to the Dana-Farber Cancer Institute.

REFERENCES

- [1] M. Bostock, V. Ogievetsky, and J. Heer. D³ data-driven documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–2309, 2011.

- [2] E. T. Brown, A. Ottley, H. Zhao, Quan Lin, R. Souvenir, A. Endert, and R. Chang. Finding waldo: Learning about users from their interactions. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1663–1672, Dec. 2014.
- [3] B. C. Cappers and J. J. van Wijk. Exploring multivariate event sequences using rules, aggregations, and selections. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):532–541, 2017.
- [4] E. Cerami, J. Gao, U. Dogrusoz, B. Gross, S. Sumer, B. Aksoy, A. Jacobsen, C. Byrne, M. Heuer, E. Larsson, et al. The cbio cancer genomics portal: an open platform for exploring multidimensional cancer genomics data. *Cancer Discovery*, 2(5):401–404, 2012.
- [5] Y. Chen, A. Puri, L. Yuan, and H. Qu. StageMap: Extracting and summarizing progression stages in event sequences. In *2018 IEEE International Conference on Big Data*, pp. 975–981, Dec. 2018.
- [6] Y. Chen, P. Xu, and L. Ren. Sequence synopsis: Optimize visual summary of temporal event data. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):45–55, Jan. 2018.
- [7] J. Gao, B. A. Aksoy, U. Dogrusoz, G. Dresdner, B. Gross, S. O. Sumer, Y. Sun, A. Jacobsen, R. Sinha, E. Larsson, et al. Integrative analysis of complex cancer genomics and clinical profiles using the cbiportal. *Science Signaling*, 6(269):p11–p11, 2013.
- [8] D. Gotz and H. Stavropoulos. DecisionFlow: Visual analytics for High-Dimensional temporal event sequence data. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1783–1792, Dec. 2014.
- [9] S. Gratzl, A. Lex, N. Gehlenborg, H. Pfister, and M. Streit. Lineup: Visual analysis of multi-attribute rankings. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2277–2286, 2013.
- [10] P. A. Greif, L. Hartmann, S. Vosberg, S. M. Stief, R. Mattes, I. Hellmann, K. H. Metzeler, T. Herold, S. A. Bamopoulos, P. Kerbs, et al. Evolution of cytogenetically normal acute myeloid leukemia during therapy and relapse: an exome sequencing study of 50 patients. *Clinical Cancer Research*, 24(7):1716–1726, 2018.
- [11] S. Guo, Z. Jin, D. Gotz, F. Du, H. Zha, and N. Cao. Visual progression analysis of event sequence data. *IEEE Transactions on Visualization and Computer Graphics*, Aug. 2018.
- [12] S. Guo, K. Xu, R. Zhao, D. Gotz, H. Zha, and N. Cao. EventThread: Visual summarization and stage analysis of event sequence data. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):56–65, Jan. 2018.
- [13] Y. Guo, S. Guo, Z. Jin, S. Kaul, D. Gotz, and N. Cao. Survey on visual analysis of event sequence data. June 2020.
- [14] J. Han, J. Pei, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M. Hsu. Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth. In *Proceedings of the 17th International Conference on Data Engineering*, pp. 215–224, 2001.
- [15] T. A. Harbig, S. Nusrat, T. Mazor, Q. Wang, A. Thomson, H. Bitter, E. Cerami, and N. Gehlenborg. Oncothreads: Visualization of large scale longitudinal cancer molecular data. In *Proceedings of the 29th conference on Intelligent Systems for Molecular Biology (ISMB)*. ISCB, 2021.
- [16] F. Inc. React.js. <https://github.com/facebook/react>.
- [17] Z. Jin, S. Guo, N. Chen, D. Weiskopf, D. Gotz, and N. Cao. Visual causality analysis of event sequence data. *IEEE Transactions on Visualization and Computer Graphics*, 2020.
- [18] B. E. Johnson, T. Mazor, C. Hong, M. Barnes, K. Aihara, C. Y. McLean, S. D. Fouse, S. Yamamoto, H. Ueda, K. Tatsuno, et al. Mutational analysis reveals the origin and therapy-driven evolution of recurrent glioma. *Science*, 343(6167):189–193, 2014.
- [19] J. Knapp, J. Zeratky, and B. Kowitz. *Sprint: How to solve big problems and test new ideas in just five days*. Simon and Schuster, 2016.
- [20] M. Krstajic, E. Bertini, and D. Keim. Cloudlines: Compact display of event episodes in multiple time-series. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2432–2439, 2011.
- [21] B. C. Kwon, V. Anand, K. A. Severson, S. Ghosh, Z. Sun, B. I. Frohnert, M. Lundgren, and K. Ng. DPVis: Visual analytics with hidden markov models for disease progression pathways. *IEEE Transactions on Visualization and Computer Graphics*, Apr. 2020.
- [22] B. C. Kwon, J. Verma, and A. Perer. Peekquence: Visual analytics for event sequence data. In *ACM SIGKDD 2016 Workshop on Interactive Data Exploration and Analytics*, vol. 1, 2016.
- [23] Z. Liu, B. Kerr, M. Dontcheva, J. Grover, M. Hoffman, and A. Wilson. Coreflow: Extracting and visualizing branching patterns from event sequences. In *Computer Graphics Forum*, vol. 36, pp. 527–538. Wiley Online Library, 2017.
- [24] L. McInnes, J. Healy, and J. Melville. UMAP: Uniform manifold approximation and projection for dimension reduction. Feb. 2018.
- [25] A. Mehrabani and B. DeLong. Intro.js. <https://github.com/usablica/intro.js>.
- [26] Mobxjs. Mobx. <https://github.com/mobxjs/mobx>.
- [27] M. Monroe, R. Lan, H. Lee, C. Plaisant, and B. Shneiderman. Temporal event sequence simplification. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2227–2236, 2013.
- [28] A. Perer and D. Gotz. Data-driven exploration of care plans for patients. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '13, pp. 439–444. Association for Computing Machinery, New York, NY, USA, Apr. 2013.
- [29] A. Perer and J. Sun. Matrixflow: temporal network visual analytics to track symptom evolution during disease progression. In *AMIA Annual Symposium Proceedings*, vol. 2012, p. 716. American Medical Informatics Association, 2012.
- [30] A. Perer and F. Wang. Frequency: interactive mining and visualization of temporal frequent event sequences. In *Proceedings of the 19th international conference on Intelligent User Interfaces*, IUI '14, pp. 153–162. Association for Computing Machinery, New York, NY, USA, Feb. 2014.
- [31] C. Plaisant, R. Mushlin, A. Snyder, J. Li, D. Heller, and B. Shneiderman. LifeLines: using visualization to enhance navigation and analysis of patient records. *Proceedings of AMIA Annual Symposium*, pp. 76–80, 1998.
- [32] M. L. i. P. scikit learn. Agglomerative clustering in sklearn. <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html>. Accessed: 2021-1-20.
- [33] E. R. Tufte. *Envisioning information*. 1990.
- [34] L. Van der Maaten and G. Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(11), 2008.
- [35] J. Walonoski, S. Klaus, E. Granger, D. Hall, A. Gregorowicz, G. Neyarapally, A. Watson, and J. Eastman. Synthea novel coronavirus (COVID-19) model and synthetic data set. *Intelligence-based Medicine*, 1:100007, 2020.
- [36] G. Wang, X. Zhang, S. Tang, H. Zheng, and B. Y. Zhao. Unsupervised clickstream clustering for user behavior analysis. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pp. 225–236, 2016.
- [37] S. Wold, K. Esbensen, and P. Geladi. Principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, 2(1-3):37–52, 1987.
- [38] K. Wongsuphasawat and D. Gotz. Exploring flow, factors, and outcomes of temporal event sequences with the outflow visualization. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2659–2668, Dec. 2012.
- [39] K. Wongsuphasawat, J. A. Guerra Gómez, C. Plaisant, T. D. Wang, M. Taieb-Maimon, and B. Shneiderman. LifeFlow: visualizing an overview of event sequences. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pp. 1747–1756. Association for Computing Machinery, New York, NY, USA, May 2011.
- [40] E. Zraggen, S. M. Drucker, D. Fisher, and R. DeLine. (s— qu) eries: Visual regular expressions for querying and exploring event sequences. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pp. 2683–2692, 2015.
- [41] Y. Zhang, K. Chanana, and C. Dunne. IDVis: Temporal event sequence visualization for type 1 diabetes treatment decision support. *IEEE Transactions on Visualization and Computer Graphics*, Aug. 2018.
- [42] J. Zhao, Z. Liu, M. Dontcheva, A. Hertzmann, and A. Wilson. Matrixwave: Visual comparison of event sequence data. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pp. 259–268, 2015.