Austrian Marshall Plan Foundation Scholarship Report

CLUE
Capture, Label, Understand, and Explain Guided Visual Explorations of Heterogeneous Data

DI Samuel Gratzl, Bsc.

Supervisor

Assist.-Prof. DI Dr.techn. Marc Streit

In Cooperation With

Pfister Lab
Harvard School of Engineering and Applied Sciences

Park Lab
Harvard Medical School

HARVARD
School of Engineering
and Applied Sciences

JOHANNES KEPLER
UNIVERSITY LINZ | JKU

HARVARD
MEDICAL SCHOOL

July 2015

Abstract

Discovering interesting findings is a challenging task. Besides the number and size of different datasets their complexity is the most demanding factor. Cancer genomics is a prime example for this. Visual analytics as the combination of data mining algorithm with human visual exploration is commonly used to tackle this challenge. However, discovering the pattern isn't the end of the analysis process. The finding needs to communicated, presented, published, and be reproducible in order to advance the own science. In this report we present CLUE. A new concept for capturing, labeling, understanding and explain visualization driven explorations. Besides the generic approach, a new open source web visualization platform Caleydo Web is introduced, implementing the CLUE concept and targeting at the biomedical domain.

Contents

List of Figures

Chapter 1

Introduction

Over the last few decades many scientific fields have been confronted with tremendous amounts of data and continuously increasing annual growth rates. Therefore, the grand challenge has shifted from the acquisition of the data to its analysis [Nie09, TC05]. Besides the sheer amount of data, particularly its complexity poses a problem for state-of-the-art analysis techniques. It is necessary to discover features and patterns across heterogeneous datasets from different sources, on distinct levels of scale, and of various types (tables, text, graphs, etc.) [KKEM10]. Integrative cancer genomics, for instance, is a prime example where analysts are confronted with such large and heterogeneous data.

While automated methods scale to large datasets, they are limited for solving knowledge discovery tasks in scenarios that include a variety of datasets that need to be investigated together. For gaining new insights, where, for instance, multiple complex relationships contribute to an effect and dominant effects can obscure weaker but highly relevant patterns, human analysts need to be included in the analysis process. Only humans, with their ability of sense-making, paired with background knowledge and intuition, can judge whether an effect is relevant in a particular context. The young science of Visual Analytics incorporates this idea.

However, new challenges arise when bringing humans back in the loop. A key point in the scientific process is reproducibility, especially in the biomedical domain. A recent review showed that it was not possible to reproduce the findings from almost 90% of over

50 cancer genomics studies [BE12]. This highlights the need for all stages of the analysis to be reproducible, interpretable, and communicable, including the visual analysis. While automated methods are easy to track, log, and therefore reproduce, humans with their creativity and mental processes are much harder to track and require advanced tools and concepts.

1.1 Approach

In this report we introduce the CLUE concept. CLUE is general applicable concept for capturing, labeling, understanding, and explaining visualization-driven explorations. Recording a provenance graph containing all actions performed during the visual analysis builds the basis of this concept. On top of it, analysts can annotate individual stages retaining their findings and explaining their decisions. Based on this annotated graph, editors can select individual key points of the analysis and form a story out of it. Finally, the resulting story can be presented, shared, and be the starting point of a new analysis.

1.2 Outlook

The remainder of this report is as follows. It starts with a introduction into the problem domain and a task analysis (Chapter 2). Then, the CLUE concept is introduced and how it relates to the author's core research topic: *Guided visual exploration of heterogeneous data* (Chapter 3). Afterwards, *Caleydo Web* is presented, a visual analytics platform for biomedical data implementing the CLUE concept (Section 4.1). Moreover, a first prototype of *StratomeX.js* is explained in detail (Section 4.2). StratomeX.js is a port of a successful visualization technique for cancer subtype analysis [SLG+14] enhanced with the CLUE concept and built using Caleydo Web. In addition, a prototype of an advanced animation concept using particles is introduced allowing tracking complex presentation changes effectively (Section 4.3). Finally, The last Chapter 5 of this report consists of a discussion, conclusion and possible future work.

Chapter 2

Background

Scientists never work alone. They are in a group with whom they discuss their findings, present their findings in a paper for the public, and use and continue the work of others. So, findings needs to be communicated, presented, understandable, and reproducible. Figure 2.1 illustrates a simple workflow of Anne, a scientist in the biomedical domain.

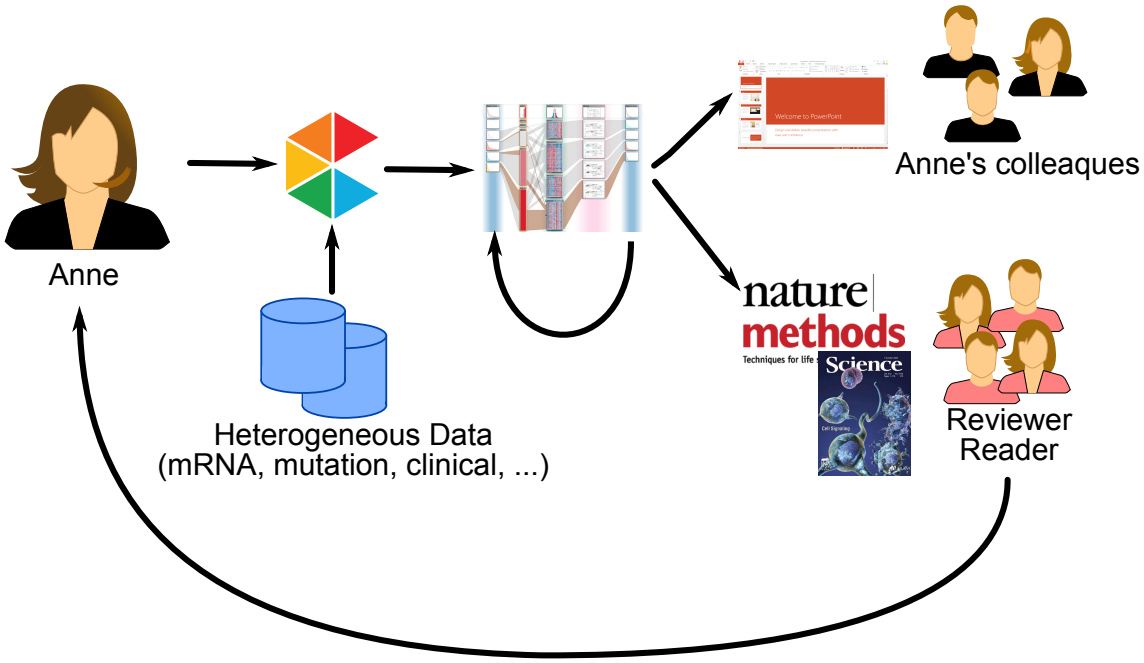

Figure 2.1: *A possible workflow in the visual analysis of biomedical data.*

She analyses her heterogeneous data from various data sources and of various types using visual analytics tools. This is an iterative process. Some of the tested hypotheses may be rejected but some may lead to interesting findings. The resulting findings needs to be presented to her colleagues, e.g., using a PowerPoint presentation. Depending on the

quality of the findings they may be published in a journal like Nature Methods. So, high quality figures with annotations explaining the findings are needed. Readers may want to work on top of the findings. Thus, a clear description how to reproduce the findings is essential.

This figure motivates the need for sophisticated tools supporting Anne in her work. The key to traceability and reproducibility lies in the collection of information about the processed data, the applied visual and computational tools, and their parameters over time. We refer to this bundle of information as **provenance graph**.

2.1 Task Analysis

Based on Figure 2.1 and our experience in the problem domain, we found following tasks that a system supporting reproducibility at all stages should support:

T I: Record Analysis Actions The key for reproducibility is recording all analysis actions performed by the users. This includes among others the selected datasets, the visualization technique to show them, and the current selection, in total the whole provenance graph.

T II: Filter Interesting Results During the analysis the user may find interesting patterns but also several uninteresting ones. A task of the user it to filter the interesting results within the provenance graph to focus on the important elements and skip the dead branches.

T III: Annotate Findings Most of the findings don't stand for them own. Users need to explain the finding by annotating them. Depending on the finding the annotation relates to a whole dataset, a collection of individual data items, a single data item, or a relation between multiple ones.

T IV: Export Findings Exporting annotated findings is an important task for communicating them. Typical examples are exports to image formats like PNG or PDF.

T VI: Share Findings Besides exporting the finding to a static snapshot in an image, the finding itself needs to be sharable. This allows other to take a look the finding and persist it along with its provenance.

T VII: Modify and Continue Findings Having a shared finding, the last task is allowing modifying and continuing the work on it. This ensures that other scientists can use the data, processing, and the finding itself for their own work, without starting from scratch.

2.2 Challenges

The defined tasks raise several challenges. Recording and tracking all actions is foremost a technical challenge, since it requires that the whole visual analytics tool is designed for it. In addition, provenance information can be collected on different levels. From user interaction and click behavior to semantic elements. For example, clicking a button can be stored as that the user clicked at the specific position on the screen or the resulting action, like adding a new dataset to the analysis. The former one is easier to track since it is a very general applicable approach. However, it provides no semantic information about the actions itself, hampering the understanding of the resulting provenance graph. The latter approach provides meaningful actions. However, they have to specialized and integrated in the system.

Exporting, sharing and modifying findings are other challenging tasks. A possible way to export the finding is not to export it as an image, but to export an embedded simplified version of the finding displayed in the tool itself. The advantages are manifold. Analysts could switch between presenting and analyzing the data easily. For example, if during the discussion of the presented the findings some new aspects come up, can they be directly tested. Individual parts of the findings could be changed on the fly during the presentation, like the selection of specific items. Animations between different steps of the analysis can be performed in a meaningful way, instead of a static series of exported images.

The pure size of the provenance graph is another challenge. Since all actions are recorded the graph grows quite fast. One challenge is how to visualize the graph itself, such that it scales to a large number of nodes but still be understandable to the user. One way is using semantic aggregation and the identification of common sub structures. For example, performing the same standard analysis steps on different datasets can be grouped together.

Finally, presenting finding in a structured and automated way is not trivial. Analysts commonly use tools like PowerPoint or KeyNote for presenting their results. However, they are designed for showing simple animations, text, and images, and not complex sequences of heterogeneous visual analyses. The balance has to be found between the simplicity and flexibility of the presentation possibilities and the feature richness a visual analytics tool should provide.

Chapter 3

CLUE Concept

The CLUE concept is a general concept for **C**apturing, **L**abeling, **U**nderstanding, and **E**xplaining visualization-driven explorations. It covers the full spectrum from visual exploration of the data to presenting the results to colleagues or other consumers. This spectrum builds an exploration-presentation continuum.

3.1 Exploration-Presentation Continuum

As described in Chapter 2 finding interesting patterns is not the end of the analysis process. Findings need to be reproducible, understandable, and communicable to others. Four different stages within the exploration-presentation continuum can be identified and are shown in Figure 3.1 with an accompanying example from the movie maker domain.

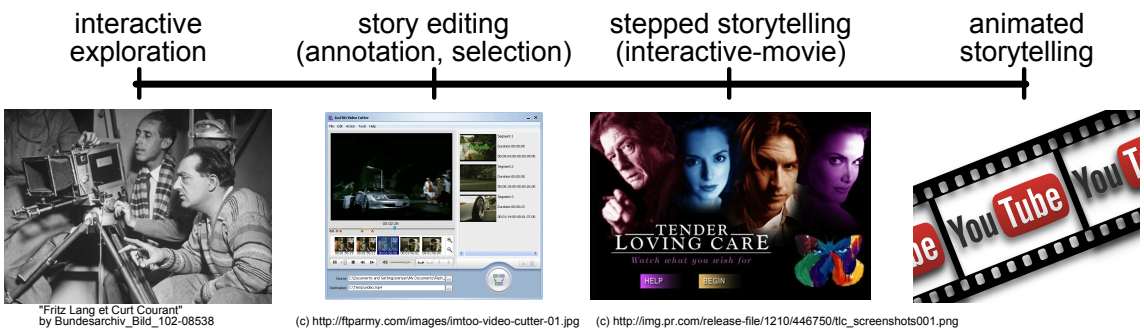

Figure 3.1: *Illustration of the exploration-presentation continuum with an accompanying example from the movie maker domain.*

EP I: Interactive Exploration This is the typical visual exploration task during an analysis. The analyst explores the data and tries to find interesting pattern, i.e., findings. Some of them are interesting some are not, but all of them are part of the analysis process. As Thomas A. Edition once said: ”I have not failed. I’ve just found 10,000 ways that won’t work.” failures are also important part of an analysis. This stage is similar to the task of a movie director. The director has the freedom to explore different setup and constellations. Some takes may fail, but you only need one good one.

EP II: Story Editing After one or more interesting findings are found the next step is to prepare them in order to present them. A story needs to be defined by selecting interesting intermediate steps for explaining the final finding. This is similar to the cutter’s task during a film production. She has to select, cut, and order individual scenes building the story of the movie.

EP III: Stepped Storytelling Presenting the findings can be done in two different favors. In stepped storytelling consumers have some degree of influence on the story. For example, they can select in which direction the presentation should continue on a branching point. Another way is altering the story during the presentation, e.g., changing the currently selected elements or the way a dataset is presented. Some modifications will influence the story itself, some won’t. Interactive movies as you may find them on some DVDs are corresponding examples in the movie maker domain. These kinds of movies allow the users to influence the movie, by letting the consumer choose between different options. A simple example is choosing between alternative endings.

EP IV: Animated Storytelling The last stage on the exploration-presentation continuum is animated storytelling. At this state the findings are presented in a fully automated way, similar to a classical movie. It may be annotated with explanations or links to further resources. Youtube videos have similar capabilities. Consumers watch them and Youtube video creators have the possibility to include popups and annotations to their movie, allowing some degree of interactivity.

The exploration-presentation continuum expresses that there is no separation between exploring the data and presenting them. Both have smooth transitions between each other. Users can explore their data and by selecting their state share and present it immediately. Similarly, when presenting some interesting findings, users can quickly jump back to exploration if some ideas pop up during discussing the results. Another aspect which is not mentioned yet, is how guidance can play a vital role within the exploration-presentation continuum.

3.2 Guided Visual Exploration of Heterogeneous Data

The main objective of the author's core research topic is to develop an interactive visualization tool for **guided exploration**, **hypotheses confirmation**, and **communication** in the area of cancer genomics. Guided exploration allows analysts to find patterns and relationships in the data that would have remained undiscovered using traditional visual data mining approaches. Hypotheses confirmation covers the important aspect of verifying analysts intentions by evidence contained in the data. Finally, the found results need to be communicable, such that other analysts can understand, use, and reproduce them. Although the research focuses on cancer genomics as application, the planned techniques will be applicable for any other domain where researchers need to make sense of large and heterogeneous data.

Guidance Continuum

Based on Schulz et al. [SSMT13] analyst's guidance types build a continuum ranging from *no guidance* to *annotated animation*. Figure 3.2 illustrates the continuum with an accompanying example from the navigation domain. The more guidance is provided to the user the less freedom she has. The following five different characteristics can be distinguished:

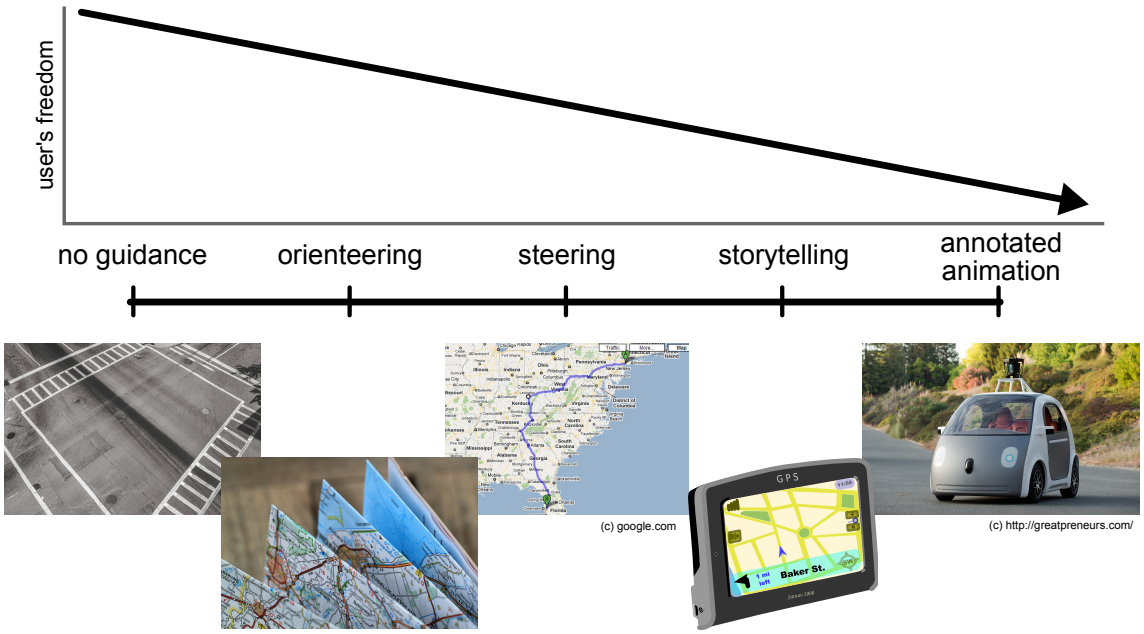

Figure 3.2: *Illustration of the guidance continuum with an accompanying example from the car navigation domain. User's freedom decreases with increasing guidance.*

G I: No Guidance The simplest guidance approach is to provide no guidance at all to the user. While this ensures that users have total freedom in their choices, they may overlook interesting patterns in their data. This is similar to drive into the blue and randomly choice a side at a crossing. While this may lead you to unexpected places, you may miss fascinating ones right around the corner.

G II: Orienteering A first guidance approach is to give the analysts orientation. This includes an overview over previous choices and future options in terms of analysis steps, data selections, or processing options. Coming back to the car navigation example this is similar to equip the driver with a map.

G III: Steering A more direct approach of guidance is to steer the analysts to interesting patterns in their data. However, in this stage the guidance is rater static, similar to a suggestion. Like looking up a route to a target, printing it out, and use it. The driver may decide to use a different route, e.g. due to a traffic jam. However, the route doesn't adapt accordingly.

G IV: Storytelling As the name suggests storytelling is about telling a story to the user and guiding her to an interesting pattern in her data. The user still has full con-

trol, however, the system can adapt to changes or different choices made by the user. This is similar to a car navigation system using GPS in which the system and guidance adapt according the actual environment and choices.

G V: Annotated Animation The other extreme end of guidance is an annotated animation in which the system takes full control over the analysis and presents interesting pattern to the user. This is similar to using a self-driving car where the user just enters the target address and the cars drive to this address autonomously.

3.3 Relevance

The CLUE concept addresses the communication aspect of the core research topic. So, it tries to answer the question how to present the findings to others efficiently and effectively. Two roles exist in the exploration-presentation continuum: **Analyst** performs then visual exploration and prepares the presentation. **Consumer** listens to the presentation and just views what the data look like. Figure 3.3 shows the relevance of individual roles within the continuum.

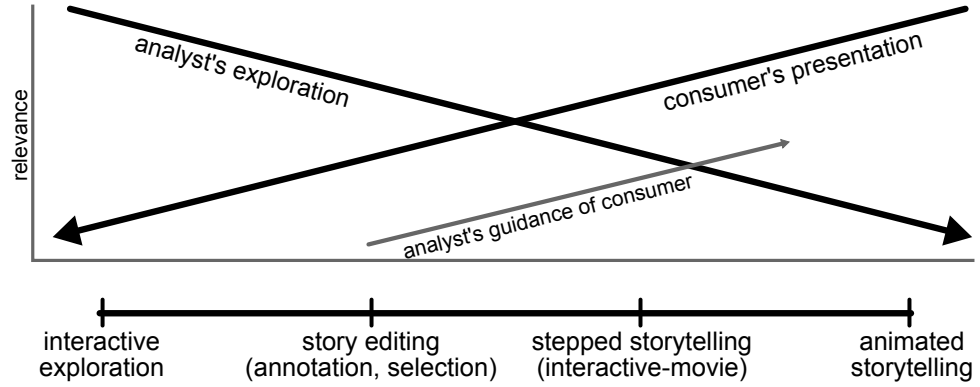

Figure 3.3: *Role (analyst, consumer) relevance within the exploration-presentation continuum. In addition, the analyst becomes more and more the guide for the consumer.*

An interesting aspect of Figure 3.3 is that the analyst becomes more and more the guide for the consumer the more the continuum shifts to the presentation stage. By selecting aspects of the visual exploration and combining them to a story the analyst preselects and guides the consumer to interesting parts. However, in this scenario a human is the guide for another human. In previous approaches automated methods and statistics were used to guide the user.

3.4 Related Work

3.4.1 Storytelling

Telling a story using visualization becomes more and more an important part of visualization research. Kosara and Mackinlay [KM13] see storytelling even as ”the next logical step”. Several papers—including [Fig14, MLF+12, LHV12, KSJ+14, GP01]— already use or promote the advantages of storytelling approaches in different domains. Even commercial visualization platforms begin to include storytelling aspects in their products. For example, Tableau [1] recently added a storytelling feature allowing creating a series of plots, annotating, and presenting them.

Wohlfart and Hauser [WH07] published a very relevant paper about ”Story Telling for Presentation in Volume Visualization”. In their work they not only promote the use of storytelling approaches for scientific volume visualization but also include concepts how and to what degree consumers can manipulate the presentation. This is similar to our exploration-presentation continuum (see Section 3.1) in which users can freely switch between exploring and presenting their data. However, in their paper they focus on a single dataset at a time, i.e. one volume visualization. In this work we focus on the presentation of multiple heterogeneous datasets, which may be partly dependent on each other.

3.4.2 Provenance Graph

Recording and exploring provenance graphs of visual analytics system is used more than a decade ago. VisTrails [BCS+05] is the most prominent example of integrating a provenance graph in a scientific visualization system. The basic approach is to record all parameter settings and user choices that lead to the current volume visualization. The resulting graph can then be used for implementing an undo-mechanism. In addition, comparing different states of the provenance graph allow to quickly investigate the effects of changed parameter settings.

[1] http://www.tableau.com/

Besides, this implementation of a provenance graph, Kreuseler et al. explore and define a history model for visual data mining [KNS04]. They extend the definition of a visual data mining system with history functionality and discuss how different operations depend on each other.

Heer et al. implement in their work on "Graphical Histories for Visualization: Supporting Analysis, Communication, and Evaluation" [HMSA08] an extension to the commercial Tableau product integrating graphical history and undo functionality. They show the graphical history using small thumbnails which the user can annotate, bookmark, and navigate to. Apart from that, the authors examine how provenance graphs can be compressed by merging nodes together. While their work is an important ground work, the CLUE concept focuses not only on the recording and exploring of provenance graph but how it can be used for presenting ones findings. To sum up, none of the existing work neither in storytelling nor provenance graph visualizations covers the full range from exploring data to presenting the found results in a similar and comprehensive way.

Chapter 4

Approach

This chapter explains how the CLUE concept is implemented. It starts with the description of a new visualization platform called *Caleydo Web*. Besides the key aspects of the platform the underlying architecture is explained in detail. Afterwards, this chapter focus on *StratomeX.js* a port of a successful existing visualization technique for cancer subtype analysis enhanced with CLUE concepts and implemented on top of Caleydo Web. The last part introduces a new animation concept and prototype using particles for easier tracking of individual elements.

4.1 Caleydo Web

Significant breakthroughs in the acquisition but also in the storage of scientific data have shifted the grand challenge in many science domains to data analysis [Nie09]. A prime example for this shift is molecular biology, where large initiatives like *The Cancer Genome Atlas* project and emerging technologies such as single cell gene sequencing produce vast amounts of heterogeneous data. Visual analysis is a key approach for making sense of the data. However, with datasets from different sources, with different meanings, on distinct levels of scale, and of various types (tables, text, graphs, etc.), there is the need for new visual analysis platforms that tackle these new challenges. *Caleydo Web* is a new open source visual analytics platform for biomedical data.

4.1.1 Key Aspects

From our experience working with domain collaborators and designing visualization systems in the past [LSKS10, LSS+12] we identified six key aspects that a visual analysis platform for biological data needs to support (see Figure 4.1):

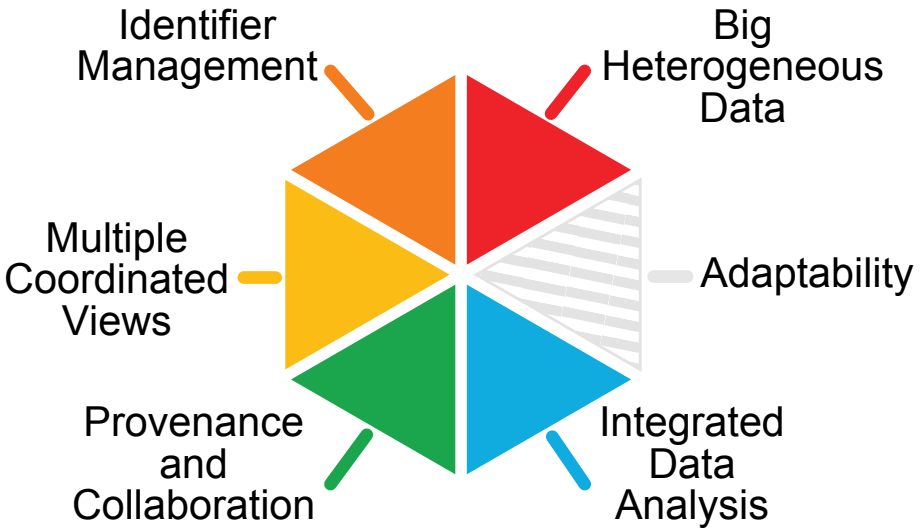

Figure 4.1: *Six key aspects of Caleydo Web*

A I: Data Scale and Heterogeneity Not only is the size of individual datasets increasing, there is also a growing number of publicly available datasets that researches want to integrate. Taken together, we observe that the size, complexity, and heterogeneity increases beyond current analysis and visualization capabilities. The data spectrum ranges from clinical and expression data, over epigenetic data, to full genome sequence information. Challenges include selecting, accessing, processing, and interactively visualizing the data.

A II: Identifier Management An important aspect when integrating datasets from various sources is the mapping of identifiers between different annotation systems (e.g., Entrez, DAVID). Mappings, however, can be 1:1, 1:*n*, *n*:*m*, or even more complex if they are based on partially overlapping gene locations. Also, entities of different types (e.g., gene, protein, samples) that can be defined on different levels of granularity (e.g., chromosome, gene, base pair) lead to additional challenges.

A III: Multiple Coordinated Views (MCV) The integrated analysis of multiple interconnected datasets can lead to new insights, yet it is often sensible to show different datasets as independent views, as the visualization can then be chosen to best represent the data. The coordination of these views provides the links between the datasets. The MCV system needs to visually link the data entities across the various annotation systems and granularity levels involved.

A IV: Provenance and Collaboration A recent review showed that it was not possible to reproduce the findings from almost 90% of over 50 cancer genomics studies [BE12]. This highlights the need for all stages of the analysis to be reproducible, interpretable, and communicable, including the visual analysis. Integrated support for provenance tracking, sharing of results, communication, and collaboration are essential.

A V: Integrated Data Analysis The integration of algorithms, statistics, and machine learning approaches like clustering or dimensionality reduction are crucial for most applications of visual analysis platforms to biomedical data. The back and forth between analysts and algorithms should be as tight and swift as possible. For instance, when a data query cannot provide immediate feedback due to the complexity of the query or the size of the data, the system should report intermediate results which the analyst can use to judge the correctness and suitability of the parameterization and adjust them if necessary [MPG+14]. Data mining algorithms can also be used for guiding analysts to interesting patterns in the data proactively [SLG+14].

A VI: Adaptability The last key aspect deals with the adaptability to changing environments. A visualization framework needs to be flexible enough to allow for, e.g., the addition of new data types, storage backends, visualization techniques, or processing algorithms. The platform should also support the creation of customized setups that are tailored to a specific application use case.

4.1.2 Related Work

BioJS

BioJS [GGS+13][1] is a library for representing biological data. Figure 4.2 shows a screenshot of their homepage. Its core is a small event-driven architecture that can be extended via plugins that are collected in a public registry. Interfaces are not defined by the library but described within a plugin's documentation only. This allows easy setup and creation of plugins for a range of different data types (A VI and A I). However, developers aiming at using multiple plugins in a setup with multiple coordinated views have to handle the synchronization and data mapping between individual plugins manually—hampering A II and A III. Moreover, the library focuses on the visualization of data only, not how it is accessed or processed (A V). Dealing with large datasets in web-based frameworks is particularly challenging, since transferring the whole dataset to the client is not an option.

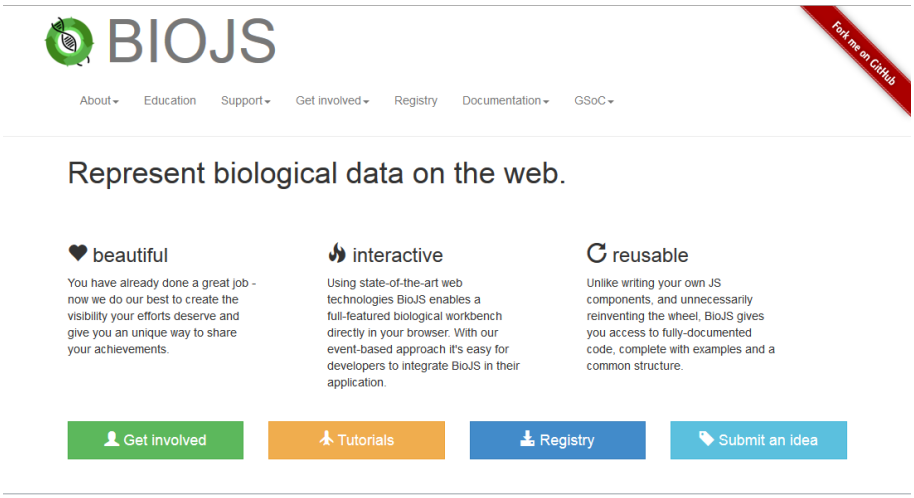

Figure 4.2: *Screenshot of the BioJS website:* `http://biojs.net`

Caleydo

Caleydo [LSKS10] is a standalone visualization framework for biological data and the predecessor of the proposed framework. Caleydo supports A II and A III, however, it lacks support for large datasets (A I), since it is a client-only application in which all datasets are loaded into main memory. Moreover, it has only rudimentary support for provenance (A IV) via a simple undo mechanism and the integrated data processing (A V)

[1] `http://biojs.net/`

is limited to a fixed set of hard coded algorithms, such as various clustering algorithms. Figure 4.3 shows a screenshot of their homepage including references to various projects implemented using Caleydo.

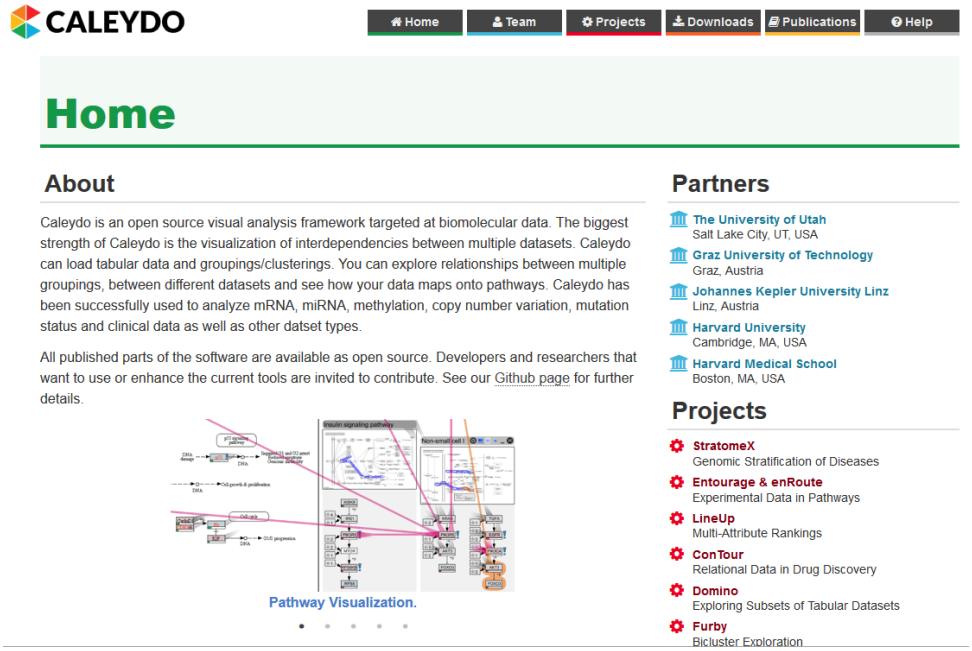

Figure 4.3: *Screenshot of the Caleydo website:* `http://caleydo.org`

4.1.3 Architecture

CLUE is based on a client-server architecture with a plugin mechanism on both sides. Client and server are coupled loosely via REST and WebSocket interfaces such that individual components can be replaced. By default, a web browser-based client and a Python server are used. Alternative possible clients include an R client for using the server API as centralized data access, or server components written in different programming languages like Java.

The plugin architecture uses a runtime environment with lazy-loaded plugins implementing extensions on one or both ends. The types of extensions include visualizations, data providers, data types, data formatters, or applications. An application is a customized and specialized arrangement of plugins for a specific purpose. For example, *StratomeX.js*, is a web-based reimplementation of the Caleydo *StratomeX* [SLG+14, LSS+12] technique. Figure 4.4 illustrates the interplay between the individual components. We are also working on a public registry in which plugins can be published, explored, and shared.

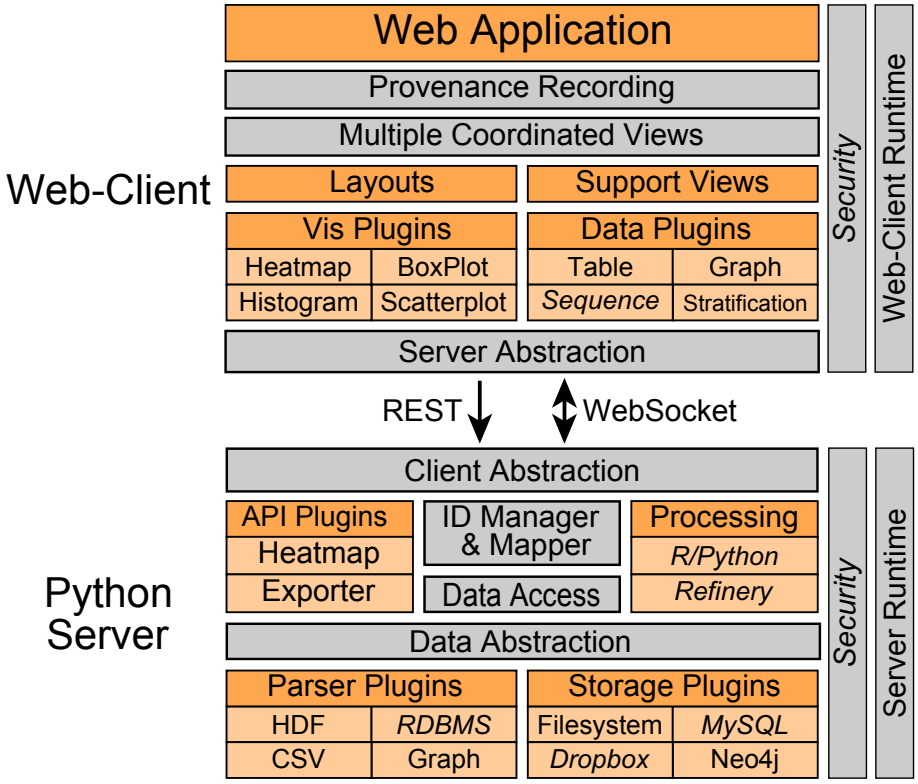

Figure 4.4: *Architectural overview of Caleydo Web. Italic labels indicate work in progress. Grey boxes represent Caleydo Web's core, dark orange boxes major plugin types, and light orange boxes custom plugins.*

This architecture allows integrating all key aspects listed in Section 4.1.1. Large data (A I) can be handled by the web-client/server architecture. Depending on the data size, only partial, aggregated, or transformed data is transferred to the user. Mapping between different annotations (A II) is implemented using a graph database. Visualization plugins select items within their dataset and the platform takes care of converting the selection into their corresponding items in other visible datasets. By using a plugin-based approach, CLUE is very flexible in terms of contained visualization techniques, dataset types, storages, and so on—addressing A VI. MCV setups (A III) are implemented by enforcing a minimal interface to visualization plugin-ins including the location of individual data points. This allows the platform to create visual links across unknown representations. A command design pattern is used for managing provenance information (A IV). For the last aspect A V, we plan to use R, Python, and Refinery[2] for executing workflows. Intermediate results and feedback on the web-client are implemented using WebSocket communication.

[2] `http://www.refinery-platform.org/`

4.1.4 Implementation

CLUE (`http://caleydo.org`) is open source under the BSD license and hosted on `https://github.com/Caleydo`. The client runtime of CLUE is implemented in TypeScript and JavaScript using HTML5. This allows visualization plugin developers to use their favorite technology, such as D3[BOH11], HTML Canvas, or WebGL. The server runtime of CLUE is implemented in Python using the Flask[3] framework. First individual plugins provide access to data storage files in HDF[4] or CSV format, and databases including Neo4j[5]. We plan to integrate an R interface for more complex data processing operations.

Registry

The plugins are organized in a public registry where they can be explored, searched, and installed. The registry is an enhanced version of the repository manager used by Node.js; NPM[6]. Individual plugins are described using a simple JSON file, called `package.json`. Besides name, version, and dependencies of the plugin, we enhance the description with entries for registering extensions points and defining external dependencies.

Plugin mechanism

Caleydo Web uses a plugin mechanism extensively. The core of Caleydo Web is just a container for managing and accessing plugins. All actual features are plugins. For example, the server implementation itself is just a plugin. This allows us to develop, improve, or replace individual plugins in a flexible way. During startup the plugin metadata are parsed and one unified plugin registry is built. Individual plugin can contribute to multiple extensions. An extension type is the placeholder for one more multiple implementations. Common extension types are visualization techniques, data storage accessors, or data parser. Each extension point description consists of the following elements:

[3] `http://flask.pocoo.org/`
[4] `http://www.hdfgroup.org/`
[5] `http://neo4j.com/`
[6] `http://npmjs.org`

- `type`

  The extension type this plugin contributes to. This is a string identifier chosen by the plugin providing the extension point.

- `id`

  A unique id identifying this extension within the extension type. This allows referring to one specific extension implementation depending on the scenario.

- `module`

  The name of the code module, i.e. the script, implementing the extension, relative to the plugin directory.

- `...`

  Besides the minimal elements for describing an extension, additional attributes can be defined depending on the extension type. For example, visualization technique extensions provide additional elements regarding the supported data types and the size of the visualization.

An important aspect of the plugin mechanism used in Caleydo Web is that code modules are just loaded when they are actually needed the first time. On the one hand, this ensures a fast startup, since only the descriptions need to be loaded and not all scripts. On the other hand, it introduces delays for loading scripts on demand during runtime. The latter one alleviates since Caleydo Web is designed for asynchronism anyhow.

External Dependencies

Besides dependencies between plugins, e.g., a visualization plugin depends on the core plugin, plugins can have dependencies to external libraries. In the current version, four different external dependency types are supported: Web, Python, Node, and Debian. During plugin resolution the individual dependencies are collected and installed.

Web Web dependencies, like javascript libraries including *D3*[7] or *JQuery*[8] are management using *Bower*[9]. Bower is a web dependency management tool allowing to quickly using web libraries. The required dependencies are defined in the `bower.json` which will be generated during dependency resolution.

Python Python dependencies, like *numpy*[10], are resolved using the *pip*[11] a python dependency manager. Similarly to Bower a special file named `requirements.txt` is generated during dependency resolution.

Node Dependencies for the Javascript server running in the Node[12] runtime are resolved using the package manager of node: NPM[13].

Debian Some external dependencies for Python or Node require that operating system specific packages are installed. For example, numpy produces errors when installed as standard dependency using pip. Debian packages are installed using the Advanced Packaging Tool (APT) [14].

Development Environment

In heterogeneous frameworks consisting of multiple components written in different programming languages, ensuring a consisted and controlled development environment is essential. In Caleydo Web we make use of *Vagrant*[15]. Vagrant is a tool for configuring virtual development environments. In our case, we set up a Debian virtual machine, in which Caleydo Web runs. The advantages compared to a local installation are manifold. A virtual machine allows a controlled environment regarding the operation system and installed packages. The local host system, i.e. the developer's machine, is not polluted by libraries needed by Caleydo Web, since all dependencies are installed just within the

[7] `http://d3js.org`
[8] `http://jquery.com`
[9] `http://bower.io`
[10] `http://www.numpy.org`
[11] `https://pip.pypa.io`
[12] `https://nodejs.org/`
[13] `http://npmjs.org`
[14] `http://wiki.debian.org/Apt`
[15] `https://www.vagrantup.com`

virtual machine. Multiple version of Caleydo Web can be checked out in parallel, since each of them has its own virtual machine. The setup of the developer environment can be automated due to the controlled environment. In the end, after checking out the repository, executing `vagrant up` will initialize the whole environment.

4.2 StratomeX.js

StratomeX.js is a Caleydo Web-based reimplementation of Caleydo *StratomeX* [LSS+12], a cancer subtype visualization technique. Figure 4.5 shows a screenshot of the application with annotations indicating individual plugins of Caleydo Web, highlighting its reusability. A demo version is available at `http://caleydo-web.herokuapp.com/stratomex.js`.

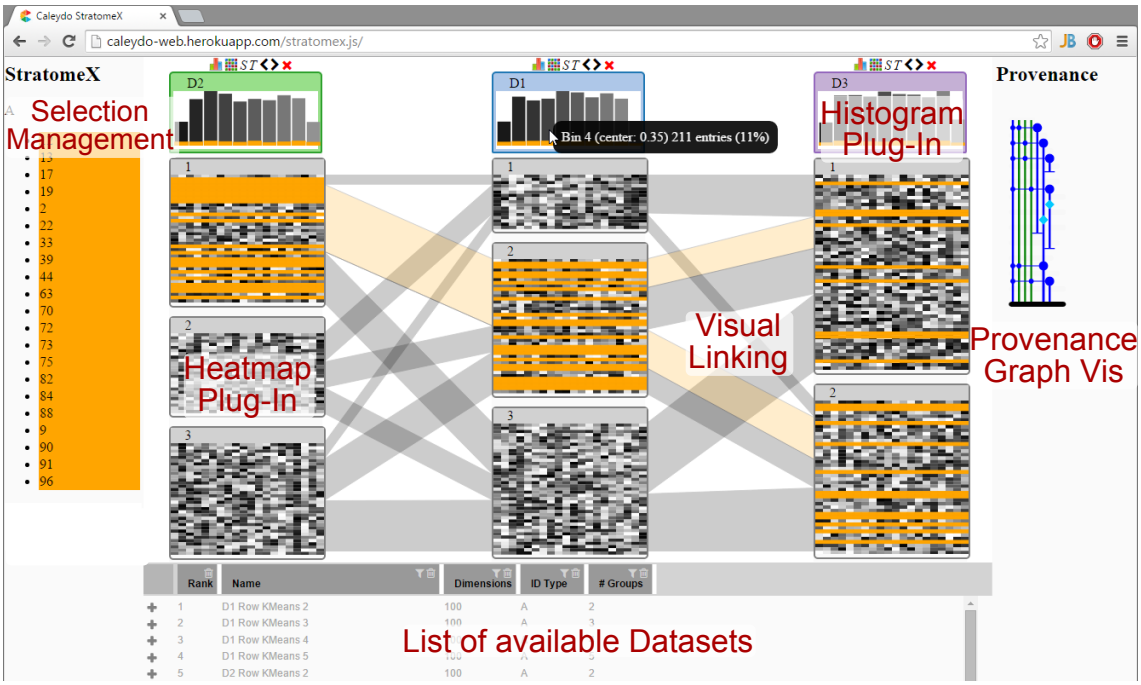

Figure 4.5: *Screenshot of StratomeX.js based on Caleydo Web. Individual components of Caleydo Web are annotated.*

4.2.1 Selection Management

The selection management consists of two parts. The visible one is shown in Figure 4.5 and consists of view showing all currently selected items. The hidden complex management takes care about the currently selected elements across multiple dataset. Individual

visualizations interact only with their corresponding dataset. So, if the user selects the third line in table A, the corresponding third row in the data table A will be selected accordingly. However, the selection management takes care of converting the third row to its corresponding row identifier, looks up all matching entries in all other visible datasets and selects them. In this example this could mean that automatically also the seventh row of table B will be selected and the corresponding visualization notified. This allows a flexible selection of items across different annotation systems and data tables.

4.2.2 Dataset Manager

The list of available datasets is presented at the bottom of Figure 4.5. Besides the name of the datasets its type, the number of items per item dimension, and the contained idtypes are shown. An idtype is a semantic concept of a data item, including patient, gene, protein, and samples. In the example figure a dummy dataset with itypes `A` and `B` is used.

4.2.3 Visualization Plugins

Individual visualizations within blocks of a StratomeX column are implemented using plugins. In Figure 4.5 two examples are given, one showing a dataset as a heatmap, one as a histogram. Users can freely choose between different visualizations for the dataset using a toolbar. Figure 4.6 shows a list of the currently available visualization plugins.

4.2.4 Visual Linking

An important aspect of StratomeX is the visual links between columns, indicating the set overlaps between individual clusters. In this example the visual linking is a general component of Caleydo Web by connecting multiple instances of the same data-item across visualizations. In this example this is not done on a per data-item level but on a more granular set level.

Due to the plugin mechanism locating data-items within visualizations is not trivial. In Caleydo Web each visualization plugin has to provide an API for accessing the position and size of a specific data-item. For example, in Figure 4.6 the heatmap plugin would

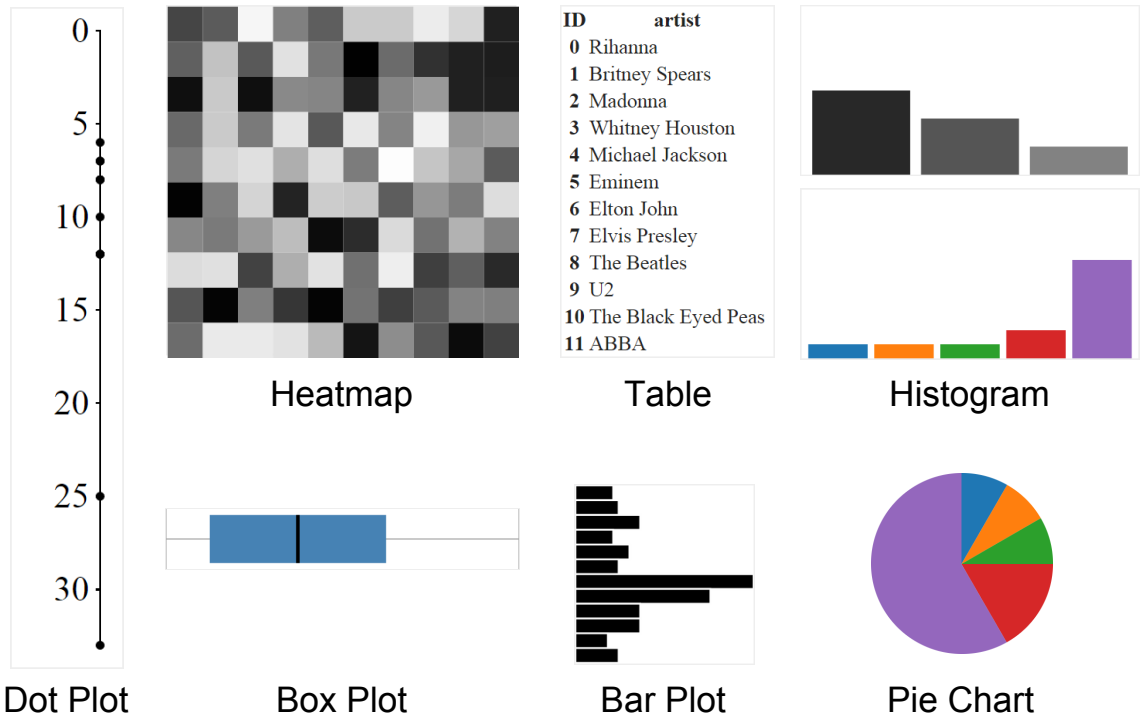

Figure 4.6: *Overview of the currently available visualization plugins in Caleydo Web: Heatmap, Dot Plot, Box Plot, Bar Plot, Table, Histogram, and Pie Chart*

return the relative position and size of a cell when requesting a combination of row and column or the size of a whole column/row if just one dimension is queried respectively. Similarly the histogram plugin would return the position and size of the corresponding histogram bin.

4.2.5 Provenance Graph Visualization

Each action within StratomeX.js is tracked and contributes to a provenance graph. In the current version we distinguish five different action categories:

Dataset Action all actions related to a dataset, like adding, subsetting, and removing them.

Visualization Action all actions about how to visualize a dataset, e.g., which visualization technique, or setting specific parameter of a technique.

Selection Action the user selections, i.e. the collection of data-items the user selects

Processing Action operations performed on the datasets, e.g. clustering algorithm.

Annotation Action all annotations added, edited, or removed from/to the analysis.

Subway Metaphor

The provenance graph visualization shown in Figure 4.5 on the right is based on a subway station metaphor. Figure 4.7 shows an example of the subway map of London. Colored tubes indicate lines and circles stations.

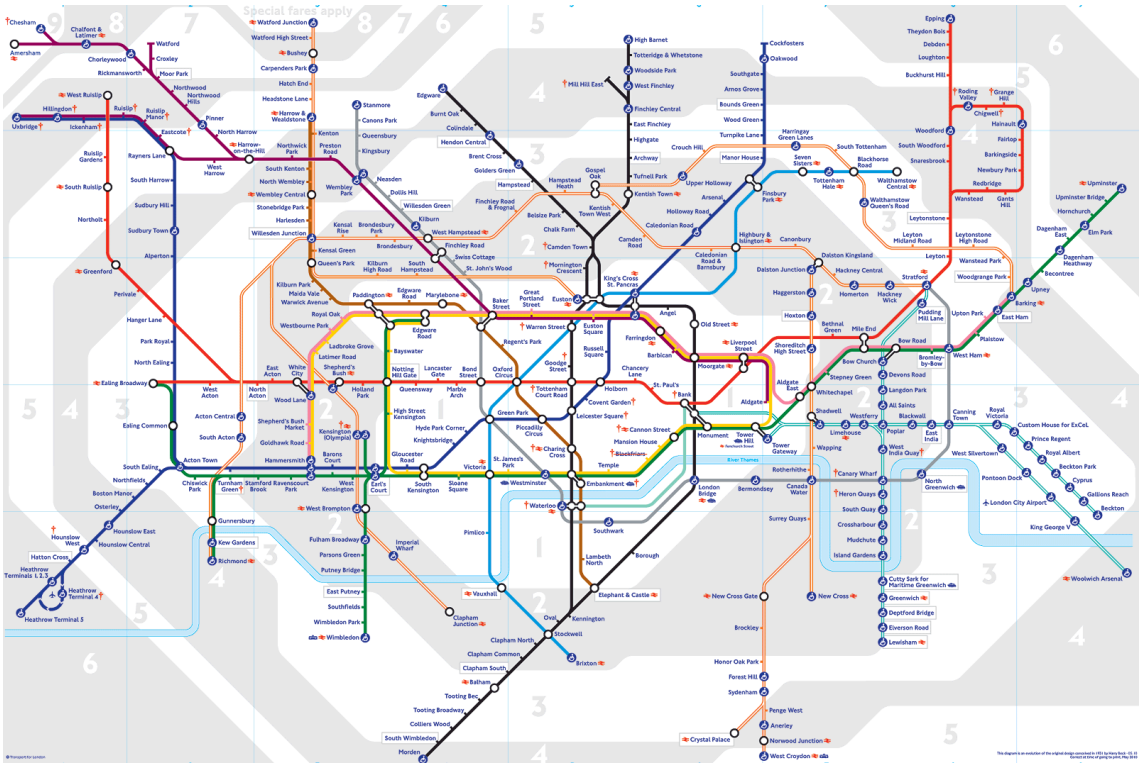

Figure 4.7: *Underground map of London used as inspiration for the provenance graph visualization*

The provenance graph visualization uses a similar idea. Lines indicate the lifetime of different items, like datasets, visualizations, or selections. For example, start and end of a line indicate adding and removing of a specific dataset respectively. Stations encode operation applied with a tube, like selecting or changing the visualization technique. Stations connection multiple tubes are actions involving multiple items at one time, like combining two datasets or creating a visualization for one dataset.

4.3 Particle Swarm

Besides the development of a new visualization platform we investigated how we can better explain how individual data items are represented in different visualizations. Bederson and Boltman[BB99] show in their user study that ”animations improves the users' ability to reconstruct the information space.” Similarly, Archambault et al. [APP11] concluded that animations are preferred when accuracy is more important than speed. So, animations are the way to go. Heer et al. show in their fundamental work on animated transitions [HR07] how different visualization techniques can be converted into other representation using animated transitions.

We decided to follow a particles idea inspired by a bee swarm. A particle represents one data-item, e.g. a patient, gene, or sample. They do not visually encode associated attributes, like patient's gender, but just a patient entity. Data are encoded by positioning particles. For example, placing particles at their corresponding position between two orthogonal axis results in a scatterplot visualization.

4.3.1 Related Work

Microsoft SandDance

Microsoft SandDance [16] is a system that uses particles for generating large scale plots. The screenshot (Figure 4.8) shows a map of the US with encoded data. However, the shape of the US is just a result of plotting individual particles at their corresponding longitude/latitude position. The map itself is not rendered.

This can also be seen in the outback of the US where the number of particles is low and therefore the shape not identifiable. The particles reflect the population density essentially. This is a general drawback of the method that is just works with a large number of particles. Using animation the particles can convert to other type of visualizations including barcharts. However, the basic approach does only work for one set of particles at a time and is not designed for aggregated visualization techniques like box plots in which

[16] `http://research.microsoft.com/en-us/projects/sanddance/`

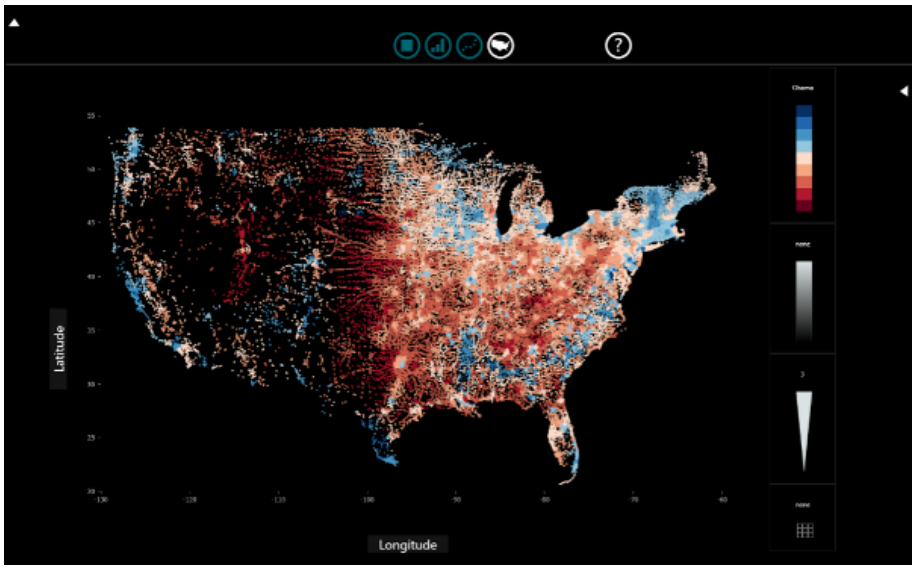

Figure 4.8: *Screenshot of Microsoft Sandance © Microsoft. The map is built from individual data particles.*

an individual data item is not represented anymore.

Visual Sedimentation

Visual sedimentation [HVF13] is an approach in which individual data elements deposit on a structure and are absorbed finally. This technique is inspired by sedimentation in nature. Figure 4.9 shows a collection of examples using the visual sedimentation approach.

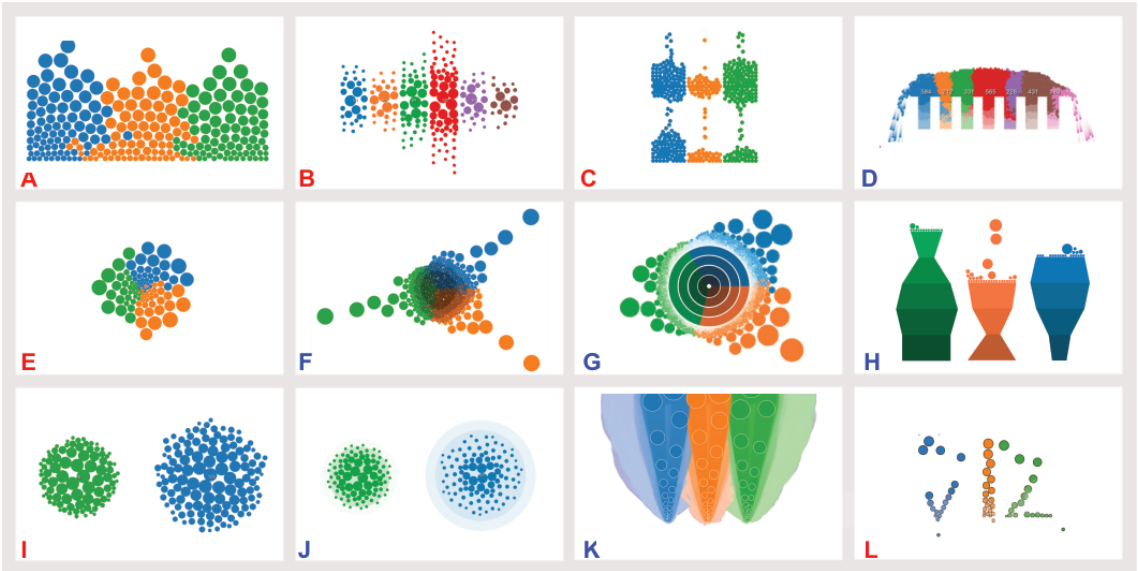

Figure 4.9: *Examples of the visual sedimentation technique [HVF13]*

Individual particles are attracted to their target position in which they finally merge to a

solid representation. However, in contrast to nature also the other way around is possible. Particles can be extracted from a solid sediment and drop off again. Visual sedimentation is useful for the visualization of real-time time-series data where you have a continuous stream of data attributes that sediment on some aggregated visualizations like a bar chart.

4.3.2 Concept

Having the idea in mind that one particle represents one data-item at a time; we played with possible animation sequences and applications of this idea. Figure 4.10 shows three sketches how a particle swarm can be used for creating three different visualizations types: scatterplot, histogram, and boxplot.

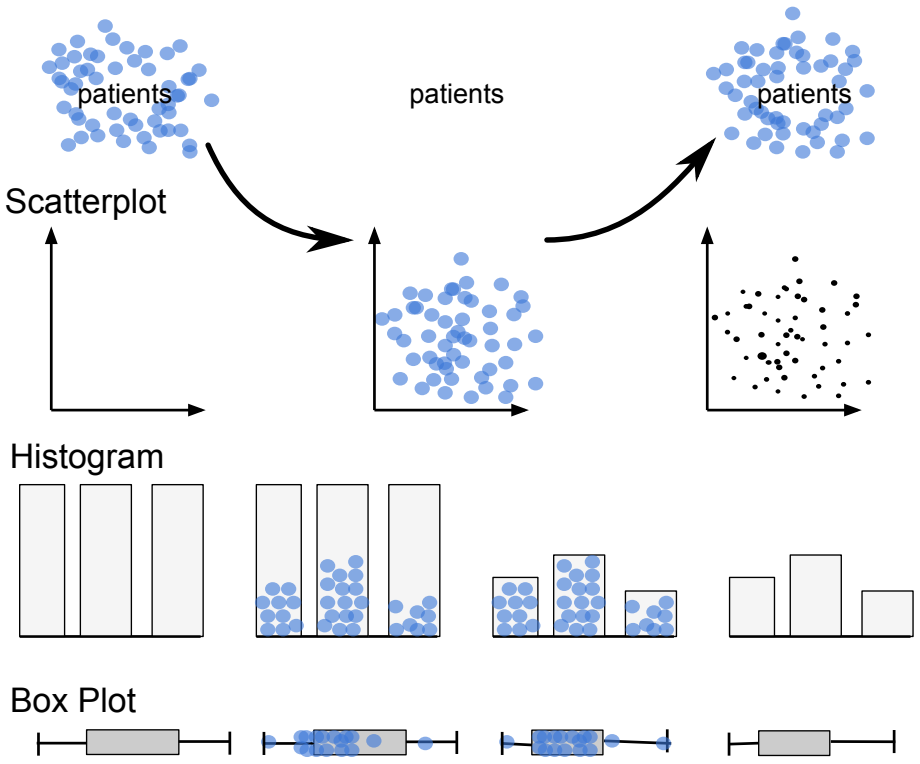

Figure 4.10: *Concept sketches of the particles idea for creating a scatterplot, histogram, and box plot based on a particle swarm*

All of them have in common that particles fan out from their hive (top left) to their position within the visualization itself and afterwards fly back to their hive again. However, depending on the visualization type different actions are performed for creating the actual visualization. Moreover, all visualizations are defined using a general frame, indicating their general setup but not encoding any data. In the scatterplot case the basic frame con-

sists of two axis and particles leave a visual mark at the corresponding position according to the given axes. Histograms and Box plots are defined by their general known structure of multiple bins and the characteristic box plot shape respectively. In these visualizations particles don't leave a visual mark but are used for defining the target position of the given frame. In the histogram case, particles sediment on their corresponding histogram bin. Afterwards, the bins are shrunk according to their actual height.

Particle Grouping

The swarm metaphor allows advanced visual operations on the set of particles itself. In Figure 4.11 a sketch is shown how a swarm of patient particles can be split up in two different swarms based on a categorical attribute. In this case patients are split up according to their associated gender. This basic idea can be generalized to other set operations including intersections and unions of different swarms.

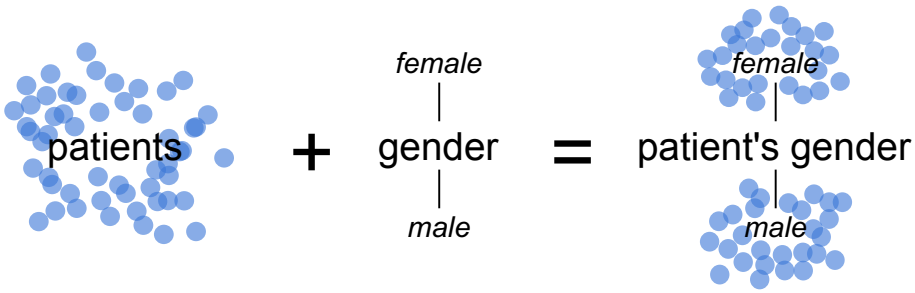

Figure 4.11: *Concept sketch of splitting a particle swarm according to a categorical attribute into multiple ones.*

Multiple Particle Types

Up to now, only one particle swarm type at a time is used. For example, Figure 4.10 shows several examples based on a single particle swarm: patients. However, this general concept allows using multiple swarms of different types at the same time. For example Figure 4.12 shows an example in which two particle swarms (blue patients and red genes) are used for building a gene expression heatmap.

The two swarms sediment on a heatmap grid. The resulting intersection points within the grid encode their corresponding value, i.e. the gene expression value of the given patient

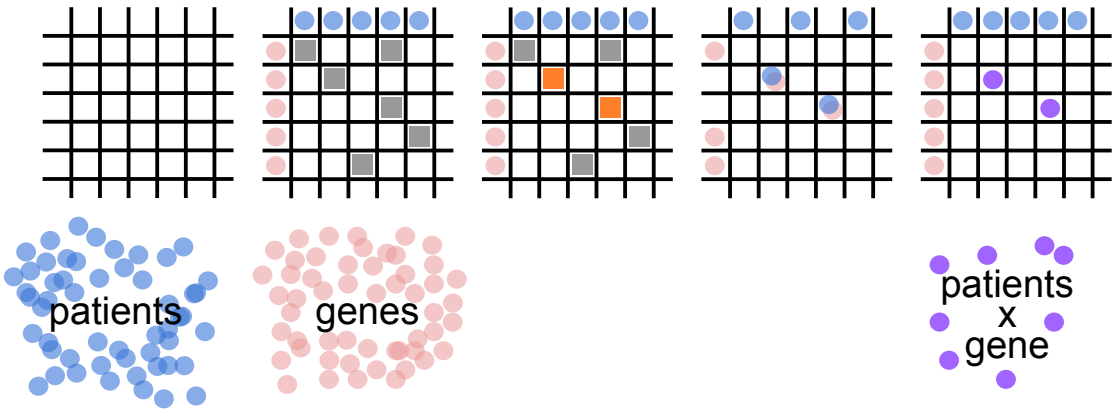

Figure 4.12: *Concept sketch of arranging two particle swarms of different types in one heatmap and building a combined new swarm type out of it.*

for a specific gene. In this example only some cells are filled. In the regular case all cells have meaningful values.

Another possible application of particle swarms is to use two types for creating a new one, by joining two particles together. This idea is also included in Figure 4.12 on the right. Based on a user selection of two cells highlighted in orange, the related particles (red and blue) join and build a new particle type (violet) representing a combination of one specific gene for one specific patient. The resulting new particle swarm can then be used in the same way as all the existing ones.

4.3.3 Examples

A first demo prototype of the particles idea is available at `http://caleydo-web.herokuapp.com/particles`. Figure 4.13 and Figure 4.14 show two image sequences of animations using the particles prototype. The former one starts with a collection of particles surrounding their hive. Moving the hive let the particles slowly follow the movement. By clicking on an axis the particles start to align them at their corresponding position according to their associated value. After the animation ends the result is a dot plot in which density indicates the number of data items at a specific value.

Figure 4.14 shows a second example of using the particles metaphor for building a histogram. By selecting the corresponding empty histogram frame for a specific attribute,

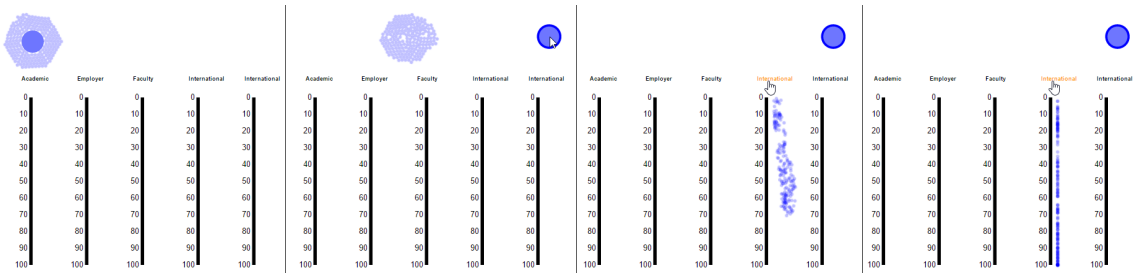

Figure 4.13: *Animation sequence showing a particle swarm deposit on an axis forming a dot plot*

the particles fly to their corresponding bin and position in a regular grid. The final heights of the histogram bins show the distribution of the data items in the selected attribute.

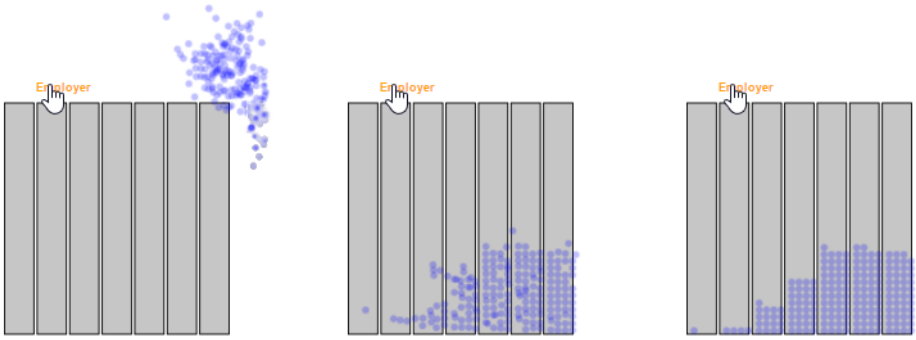

Figure 4.14: *Animation sequence showing a particle swarm deposit on empty bins forming a histogram*

4.3.4 Discussion

While the general approach looks fancy, the practical applications are limited. Scalability is a major issue. On the one hand, the visual impact of potential millions of particles. On the other hand, the computational effort for computing the position and movement of particles. While static positions and animated transitions between them are easier to compute, organic movements are very complex. One approach is using a physic simulation in the background in which individual particles repulse from each other and are attracted to their target position. However, the complexity for computing the physical simulation is enormous. In the worst case, every particle has to be compared with every other particle for computing a proper repulsion. Using advanced techniques like quadtrees can speed up the computation but the general complexity remains.

Another problem is duplicity and linking between particles. If the same data-item is visualized multiple times particles need some way to be cloned since one particle represents an individual data-item. Two different approaches are tested. The first one uses real clones of particles, such that multiple versions of the same particles exists. The second approach allows only one particle at a time but particles can leave static marks at visualizations. The former one has the advantage that the concept of a particle remains the same across all visualizations. The disadvantage is the management of multiple clones, e.g., which clone should be used for creating another visualization, how to merge clones, and so on. The advantage of the second approach is that there is always just a single instance representing a data-item. However, the particles have no direct impact on the visualizations but just their marks. This hampers the animation from one visualization to another one.

In the end, using particles for representing individual data-items results in fancy animations but have only limited practical use in a standalone version. One exception is using particles for positioned attributes, like geo locations on a map. We are currently investigating how the particles idea can be adapted for improving the transitions between different story steps in CLUE.

Chapter 5

Conclusion

This last chapter is used for reflecting the work described in this report. Besides, a discussion about the introduced concepts and approaches, possible future work is discussed. The end of this chapter and report is a short summary.

5.1 Discussion

Several aspects of this work can be discussed. In this report only two aspects (CLUE and Caleydo Web) are discussed in detail:

5.1.1 CLUE

Capture Semantic Provenance

A general problem when capturing the actions performed during the analysis is the semantic level that will be captured. Low level events like user clicks are easy to track and can be done in a generic way for different visual analytics tools. However, click traces have only limited information about what the action is about e.g., clicking a button at position (x, y) doesn't provide the information about the effects of this click. For example, it could be a selection of a data-item or adding a new dataset to the analysis. This hampers interpreting the provenance graph drastically. Further, the provenance graph grows very fast when capturing these low level actions. So, a higher level needs to be captured. However, this requires a deep integration in the visual analysis tool that can only be ensured for tools created with CLUE from scratch.

Replay Actions

Besides recording the actions taken by the user, replaying isn't trivial either. Actions have to be extracted to commands which can be triggered automatically. However, replaying all actions from the start of the analysis to get the final state isn't effective. A combination of snapshots of the whole current state with incremental actions may be a possible tradeoff.

Another challenge is the changing environments. Especially in visualizations the available screen size can influence the presentation tremendously. Different screen sizes, program versions, available resources need to be taken into account.

State Transition

Users can select one or more states of the provenance graph to create a story out of it. The resulting annotated story is then the basis for a presentation. A challenging task is how to implement the state transitions between the states. The simplest approach is using hard cuts, i.e. no transition at all and just show the following state. While this is commonly done when having a series of screenshots, it is difficult to track the changes and keep the relation between individual states. The other extreme approach is to lookup the path connecting both states in the provenance graph and replay the actions along the path. While this ensures a smooth transition, it may require several actions to be executed that need time. A possible way for speeding the transition up is to eliminate unnecessary steps, e.g., adding datasets which are removed afterwards in the path. Another way is trying to parallelize actions, e.g. changing multiple parameters at the same time. Both approaches require that the provenance graph is rewritten and modified partially. This requires a deep understanding of the individual actions and how they relate to each other.

Figure 5.1 shows an example of a provenance graph with four selected story points (a). (b) to (d) show possible animation paths. In (b) all intermediate steps are executed and partially reverted for getting from *3* to *4*. In (c) a hard cut between *3* and *4* is performed, reducing the number of animation steps. Finally in (d) an alternative hard cut is performed in which the same intermediate story point *1* is visited twice as a common starting point.

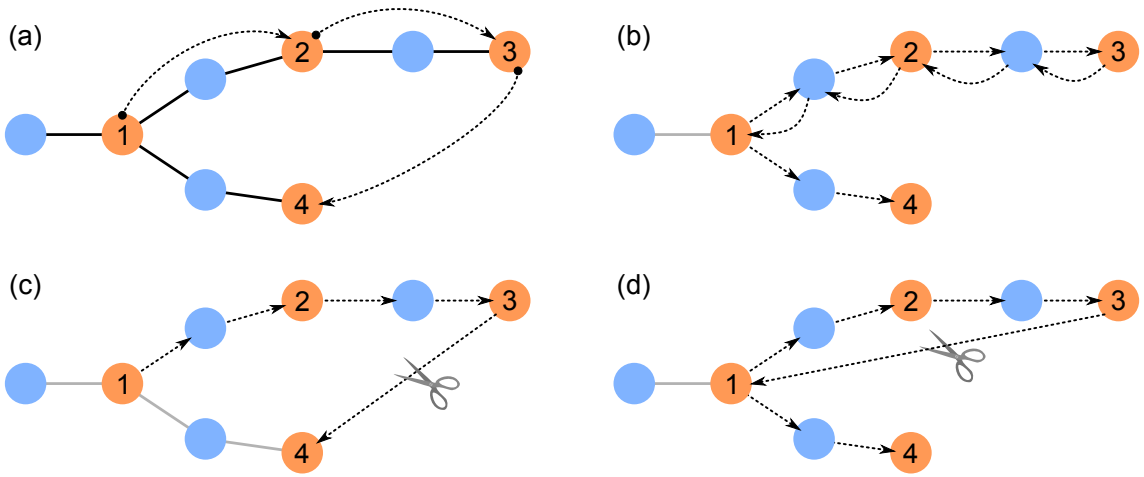

Figure 5.1: *Example provenance graph (a) with four selected story points. (b) to (d) show alternative animation paths. (b) reverts all actions and replays them while the other use hard cuts instead.*

Using particles for animation is another approach for explaining how individual items evolve and relate within different visualizations. As explained in Section 4.3 a particle represents a single data-item, e.g. a patient or a gene. Particles move from one representation into another allow tracking individual items across visualizations. However, the scalability of this approach is limited. If organized in an organic way it requires heavily computational effort. Moreover, aggregated visualizations like box plots are problematic since individual items aren't shown anymore.

Interactive Story Manipulation

A key idea of the CLUE concept is that analysis and presentation are interleaved and just two extreme of the same continuum - see Section 3.1. Therefore, users can move along the continuum smoothly. Difficulties arise when the selected and prepared story should be changed on the fly. While some changes won't affect the story, some will. This heavily depends on the story to tell. For example, changing the number of bins in a histogram doesn't influence the result, if the dataset is not further used. However, if a specific bin is extracted, changing the number of bins prohibits the rest of the story to understand, since the extraction doesn't make any sense anymore.

It depends on the actual situation whether one change is a modification of the current state or the start of a new analysis branch. On the one hand, some modifications like a new selection of items can be applied on top of the existing one and won't affect the story. On the other hand, adding a new dataset or removing an existing one, will change the story potentially. In the end the system can try to detect whether the current change will break the story and warn the user that her modification will create a new analysis branch. However, this is not a trivial task either.

5.1.2 Caleydo Web

Scalability

Scalability is a common issue for all platforms. How does the platform scale to a large number of user, data items .... The number of users is not an issue for Caleydo Web through its target audience. The target audience is a group of biologist and bioinformaticians working together in a team. Therefore, we expect a total number of less than 30 users at a time per instance. This small number of active users can still be handled by a single server without the need for load balancing.

Caleydo Web ensures scalability in the number of data items by several measures. First, it uses a client-server architecture allowing that the whole data storage is centralized on a powerful server. This avoids that all the data needs to be hosted and transferred to each client. The idea is to just transfer the data the client currently needs. Depending on the visualization it can be the raw data or aggregated and transformed versions of it. For example, for rendering a boxplot only a few statistical measures are needed instead of the whole dataset. A different option is to render a preview of the visualization on the server and deliver this non-interactive image till the real visualization has been loaded. This ensures responsiveness of the system and fast results. A common example is rendering a heatmap on the server in a image texture. Depending on the available screens space and data size this may include data sampling and grouping. The resulting heatmap texture is then used as a background image on which the current selection is added as an additional layer.

Asynchronicity and Delays

Transferring data to the client, loading plugins, and computing visualizations takes time. Therefore, asynchronous operations are an important aspect of Caleydo Web. Placeholders, cached previews, and meta information till the final result is available are countermeasure for handling delays due to asynchronous operations. Another aspect of Caleydo Web is automatic conversion between different annotation system like Entrez or DAVID. This ensures that data items selected in one annotation system are also highlighted in datasets using a different one. However, due to possible server lookup operations this may create delays between the selection of one element in a dataset and the highlighting in all related ones. Besides caching of mappings, user feedback has to be given such that she will be notified about possible delays.

5.2 Future Work

Besides continuing the ongoing work, several interesting aspects of the CLUE concept can be thought of:

The first one is to integrate guidance at all stages. Guidance at the exploration level can help analysts identifying more findings faster. The main challenges are that the user shouldn't be manipulated in her choice but just guided. You need to avoid that the analyst stops thinking and just relies on the suggestions. Guidance on the story editing level is about identifying interesting sub results and how to reproduce them automatically. Another possible way of guidance is helping the user what is not interesting at all and therefore reduce the size of the provenance graph dramatically.

Another possible future work is about analyzing the provenance graph itself. Based on one or multiple provenance graphs of multiple session and users, fascinating observations could be made. For example, like commonly repeated pattern which are executed over and over again, loops within the graph, in which users come back to the same state where they started, and so on. Furthermore, analyzing provenance graphs of visualization-driven exploration can help improving how users are guided. In the current version most guid-

ance is based on statistical measure to identify interesting patterns in the data. Another possible way is providing guidance based on existing provenance graphs and the extracted pattern within them.

Finally, since Caleydo Web is plugin based, extensions are improvements are easily possible. One long time goal is to build a community around it, such that bioinformaticians from all over the work contribute and extend the platform with their own data, visualization techniques, and applications. This would not only speed up the development of Caleydo Web but also increase the number of users, its variety, and application scenarios.

5.3 Summary

In this report ongoing work on the CLUE concept is introduced. It is a general concept for capturing, labeling, understanding, and explaining visualization-driven explorations. Besides this theoretical concept the Caleydo Web visualization platform is explained in detail including its key aspects and architecture. Particles were an attempt for realizing animations within visual analysis ensuring traceability of individual elements during the analysis. Finally, StratomeX.js, a port of an existing visualization technique, demonstrated the applicability and possibilities of the CLUE concept and Caleydo Web. The current state already shows promising results which are improved in the future.

Appendix A

Acknowledgments

I would like to thank the Austrian Marshall Plan Foundation for giving me the opportunity to visit Harvard University by granting me a scholarship. My special thank goes to Thomas Mahringer from the International Office at JKU, who is my contact to the Austrian Marshall Plan Foundation and helped with the application and all other matters I had during the preparation of my visit.

Furthermore, I would like to thank my supervisor Marc Streit, who is always a great help from writing grants and reports to brainstorming ideas when I'm stuck. Since we work together, I have learned a lot during the last years. From day zero on, he integrated me in the Caleydo project as I was always part of it. Together, we achieved to publish award winning papers like LineUp [GLG+13] or Domino [GGL+14]. I'm looking forward what the future will bring.

In addition, my thank goes to Hanspeter Pfister, who invited me to Harvard University and gave me the opportunity to work at one of the best universities in world. I would also like to thank my Harvard colleagues and friends particularly: Hendrik Strobelt, Johanna Beyer, James Tompkin, Daniel Haehn, Nils Gehlenborg, and Alexander Lex, who integrated me immediately, showed me everything, helped and discussed with me. Thanks for the warm welcome guys. It was the coldest, snowiest and worst winter Boston ever had, but one of the best times in my present life.

This work was supported in part by the Austrian Research Promotion Agency (840232), the Austrian Science Fund (J 3437-N15), the Air Force Research Laboratory and DARPA grant FA8750-12-C-0300, and the United States NIH/National Human Genome Research Institute (K99 HG007583).

Bibliography

[APP11] D. Archambault, H. Purchase, and B. Pinaud. Animation, Small Multiples, and the Effect of Mental Map Preservation in Dynamic Graphs. *IEEE Transactions on Visualization and Computer Graphics*, 17(4):539 –552, April 2011.

[BB99] B.B. Bederson and A. Boltman. Does animation help users build mental maps of spatial information? In *1999 IEEE Symposium on Information Visualization, 1999. (Info Vis '99) Proceedings*, pages 28–35, 1999.

[BCS+05] L. Bavoil, S.P. Callahan, C.E. Scheidegger, H.T. Vo, P.J. Crossno, C.T. Silva, and J. Freire. VisTrails: Enabling Interactive Multiple-View Visualizations. In *Proceedings of the IEEE Conference on Visualization (VIS '05)*, pages 135–142. IEEE, 2005.

[BE12] C. Glenn Begley and Lee M. Ellis. Drug development: Raise standards for preclinical cancer research. *Nature*, 483(7391):531–533, March 2012.

[BOH11] M. Bostock, V. Ogievetsky, and J. Heer. D3: Data-Driven Documents. *IEEE Transactions on Visualization and Computer Graphics (InfoVis '11)*, 17(12):2301–2309, 2011.

[Fig14] A. Figueiras. How to Tell Stories Using Visualization. In *2014 18th International Conference on Information Visualisation (IV)*, pages 18–18, July 2014.

[GGL+14] Samuel Gratzl, Nils Gehlenborg, Alexander Lex, Hanspeter Pfister, and Marc Streit. Domino: Extracting, Comparing, and Manipulating Subsets across Multiple Tabular Datasets. *IEEE Transactions on Visualization and Computer Graphics (InfoVis '14)*, 20(12):2023–2032, 2014.

[GGS+13] John Gómez, Leyla J. García, Gustavo A. Salazar, Jose Villaveces, Swanand Gore, Alexander García, Maria J. Martín, Guillaume Launay, Rafael Alcántara, Noemi del Toro, Marine Dumousseau, Sandra Orchard, Sameer Velankar, Henning Hermjakob, Chenggong Zong, Peipei Ping, Manuel Corpas, and Rafael C. Jiménez. BioJS: an open source JavaScript framework for biological data visualization. *Bioinformatics*, 29(8):1103–1104, April 2013.

[GLG+13] Samuel Gratzl, Alexander Lex, Nils Gehlenborg, Hanspeter Pfister, and Marc Streit. LineUp: Visual Analysis of Multi-Attribute Rankings. *IEEE Transactions on Visualization and Computer Graphics (InfoVis '13)*, 19(12):2277–2286, 2013.

[GP01] Nahum Gershon and Ward Page. What storytelling can do for information visualization. *Communications of the ACM*, 44(8):31–37, 2001.

[HMSA08] Jeffrey Heer, Jock Mackinlay, Chris Stolte, and Maneesh Agrawala. Graphical Histories for Visualization: Supporting Analysis, Communication, and Evaluation. *IEEE Transactions on Visualization and Computer Graphics (InfoVis '08)*, 14(6):1189–1196, 2008.

[HR07] J. Heer and G. G Robertson. Animated Transitions in Statistical Data Graphics. *IEEE Transactions on Visualization and Computer Graphics (InfoVis '07)*, 13(6):1240–1247, 2007.

[HVF13] Samuel Huron, Romain Vuillemot, and J.-D. Fekete. Visual sedimentation. *Visualization and Computer Graphics, IEEE Transactions on*, 19(12):2446–2455, 2013.

[KKEM10] Daniel A Keim, Joern Kohlhammer, Geoffrey Ellis, and Florian Mansmann, editors. *Mastering the Information Age - Solving Problems with Visual Analytics*. Eurographics, Goslar, Germany, 2010.

[KM13] Robert Kosara and Jock Mackinlay. Storytelling: The next step for visualization. *Computer*, (5):44–50, 2013.

[KNS04] M. Kreuseler, T. Nocke, and H. Schumann. A History Mechanism for Visual Data Mining. In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis '04)*, pages 49–56. IEEE, 2004.

[KSJ+14] Bum Chul Kwon, Florian Stoffel, Dominik Jäckle, Bongshin Lee, and Daniel Keim. VisJockey: Enriching Data Stories through Orchestrated Interactive Visualization. 2014.

[LHV12] Endre M. Lidal, Helwig Hauser, and Ivan Viola. Geological storytelling: graphically exploring and communicating geological sketches. In *Proceedings of the International Symposium on Sketch-Based Interfaces and Modeling*, pages 11–20. Eurographics Association, 2012.

[LSKS10] Alexander Lex, Marc Streit, Ernst Kruijff, and Dieter Schmalstieg. Caleydo: Design and Evaluation of a Visual Analysis Framework for Gene Expression Data in its Biological Context. In *Proceeding of the IEEE Symposium on Pacific Visualization (PacificVis '10)*, pages 57–64. IEEE, 2010.

[LSS+12] Alexander Lex, Marc Streit, Hans-Jörg Schulz, Christian Partl, Dieter Schmalstieg, Peter J. Park, and Nils Gehlenborg. StratomeX: Visual Analysis of Large-Scale Heterogeneous Genomics Data for Cancer Subtype Characterization. *Computer Graphics Forum (EuroVis '12)*, 31(3):1175–1184, 2012.

[MLF+12] Kwan-Liu Ma, Isaac Liao, Jennifer Frazier, Helwig Hauser, and Helen-Nicole Kostis. Scientific Storytelling Using Visualization. *IEEE Computer Graphics and Applications*, 32(1):12–19, January 2012.

[MPG+14] Thomas Mühlbacher, Harald Piringer, Samuel Gratzl, Michael Sedlmair, and Marc Streit. Opening the Black Box: Strategies for Increased User Involve-

ment in Existing Algorithm Implementations. *IEEE Transactions on Visualization and Computer Graphics (VAST '14)*, 20(12):1643–1652, 2014.

[Nie09] Michael Nielsen. A guide to the day of big data. *Nature*, 462(7274):722–723, 2009.

[SLG+14] Marc Streit, Alexander Lex, Samuel Gratzl, Christian Partl, Dieter Schmalstieg, Hanspeter Pfister, Peter J. Park, and Nils Gehlenborg. Guided visual exploration of genomic stratifications in cancer. *Nature Methods*, 11(9):884–885, 2014.

[SSMT13] Hans-Jörg Schulz, Marc Streit, Thorsten May, and Christian Tominski. Towards a Characterization of Guidance in Visualization. In *Poster Compendium of the IEEE Conference on Information Visualization (InfoVis '13)*. IEEE, 2013.

[TC05] James J Thomas and Kristin A Cook. *Illuminating the Path: The Research and Development Agenda for Visual Analytics*. IEEE Computer Society Press, Los Alamitos, CA, USA, 2005.

[WH07] Michael Wohlfart and Helwig Hauser. Story Telling for Presentation in Volume Visualization. In *Proceedings of the 9th Joint Eurographics / IEEE VGTC Conference on Visualization*, EUROVIS'07, pages 91–98, Aire-la-Ville, Switzerland, Switzerland, 2007. Eurographics Association.