Multiscale Visualization Using Data Cubes

A 2002 Paper by Chris Stolte, Diane Tang, and Pat Hanrahan

Presented by Curran Kelleher
Abstract

Most analysts start with an overview of the data before gradually refining their views to be more focused and detailed. Multiscale pan and zoom systems are effective because they directly support this approach. However, generating abstract overviews of large datasets is difficult, and most systems take advantage of only one type of abstraction: visual abstraction. Furthermore, these existing systems limit the analyst to a single zooming path on their data and thus a simple set of abstract views.

This paper presents: (1) a formalism for describing multiscale visualizations of data cubes with both data and visual abstraction, and (2) a method for independently zooming along one or more dimensions by traversing a zoom graph with nodes at different levels of detail. As an example of how to design multiscale visualizations using our system, we describe four design patterns using our formalism. These design patterns show the effectiveness of multiscale visualization of general relational databases.
This paper presents:

1. A formalism for describing multiscalar visualizations of data values with both data and visual abstraction.

2. A method for independently zooming along one or more dimensions by traversing a zoom graph with nodes at different levels of detail.
Abstract

Most analyses start with an overview of the data before gradually refining their view to be more focused and detailed. Multiscale path and zoom systems are effective because they directly support this approach. However, generating abstract overviews of large data sets is difficult, and most systems take advantage of only one type of abstraction: visual abstraction. Furthermore, these existing systems limit the analyst to a single, recurring path on their data and thus a single set of abstract views.

This paper presents:

1. A formalism for describing multiscale visualizations of data cubes with both data and visual abstractions, and
2. A method for independently zooming along one or more dimensions by traversing a zoom graph with nodes at different levels of detail.

As an example of how to design multiscale visualizations using our system, we describe four design patterns using our formalism.
two types of abstraction

_data abstraction and visual abstraction._
Data Abstraction
1 Introduction

When exploring large datasets, analysts often work through a process of "overview first, zoom and filter, then details on demand" ([14]). Multiscale visualizations are an effective technique for facilitating this process because they change the visual representations to present the data at different levels of abstraction as the user pans and zooms. At a high level, because a large amount of data needs to be displayed, it is highly abstracted. As the user zooms in, the data density decreases and thus more detailed representations of individual data points can be shown.

The two types of abstraction performed in these multiscale visualizations are data abstraction and visual abstraction. Data abstractions (e.g., aggregation or selection) change the underlying data before mapping them to visual representations. Visual abstractions change the visual representations of data points (but not the underlying data itself) to provide more information as the user zooms in, e.g., an image may morph from a simplified thumbnail to a full-scale...
In this section, we review several existing multiscale visualization systems, focusing on how the systems perform both data and visual abstraction.

**Data abstraction** refers to transformations applied to the data before being visually mapped, including aggregation, filtering, sampling, or statistical summarization.

**Visual abstraction** refers to abstractions that change the visual representation (e.g., a circle at an overview level versus a text entry at a detailed level), change how data is encoded in the visual attributes of the glyph (e.g., encoding data in the size and color of a glyph only at detailed views), or apply transformations to the set of visual representations (e.g., combining glyphs that overlap).

**Multiscale Visualization in Cartography**

Cartography is the source of many early examples of multiscale visualizations. Cartographic generalization [19] refers to the process of generating small-scale maps by simplifying and abstracting large-scale maps to maintain and reveal the essence of the data. Cartography can be used as the basis for creating multiscale visualizations.
Data cubes are a commonly accepted method for abstracting and summarizing relational databases.
Data Abstraction: Data Cubes

Data cubes categorize information into two classes: dimensions and measures,
For example, U.S. states are a dimension, while the population of each state is a measure.
data is abstractly structured as an n-dimensional data cube. Each axis corresponds to a dimension in the data cube and consists of every possible value for that dimension. For example, an axis corresponding to states would have fifty values, one for each state.
variables, respectively. For example, U.S. states are a dimension, while the population of each state is a measure. Within a cube, the data is abstractly structured as an n-dimensional data cube. Each axis corresponds to a dimension in the data cube and consists of every possible value for that dimension. For example, an axis corresponding to states would have fifty values, one for each state. Every “cell” in the data cube corresponds to a unique combination of values for the dimensions.

Each cell contains one value per measure of the data cube.
Projecting a three dimensional data cube

A two dimensional projection:
Projection onto (Time, Products, *)

A one dimensional projection:
Projection onto (Time, *, *)
Projection onto (*, *, *)
Thus far, we have considered dimensions to be flat structures. However, most dimensions have a hierarchical structure.

For example, rather than having a simple dimension “state”, we may have a hierarchical dimension “locations” that has levels for country, state, and county. If each dimension has a hierarchical structure, then the data must be structured as a lattice of data cubes, where each cube is defined by the combination of a level of detail for each dimension.

Data abstraction in this model means choosing a meaningful summary of the data. Choosing a data abstraction corresponds to choosing a particular projection in this lattice of data cubes (to which dimensions we currently consider relevant and to the appropriate level of detail for each relevant dimensional hierarchy). Specifying the level of detail identifies the cube in the lattice, while the relevant dimensions identifies which projection (from a dimension down to the number of relevant dimensions) of the cube is needed. Figure 1 shows a simple lattice and projections.

While identifying a specific projection in the data cube corre...
For example, rather than having a single dimension “state”, we may have a hierarchical dimension “location” that has levels for country, state, and county.
If each dimension has a hierarchical structure, then the data must be structured as a lattice of data cubes.
The lattice of data cubes

Least detailed

type
market
quarter

type
market
month

state
quarter

product
market
quarter

product
state
quarter

product
state
month

Most detailed
each cube is defined by the combination of a level of detail for each dimension.
Data abstraction in this model means choosing a meaningful summary of the data.
Choosing a data abstraction corresponds to choosing a particular *projection* in this lattice of data cubes:
Projecting a three dimensional data cube

- **Time:** qtr1, qtr2, qtr3, qtr4
- **Location:** east, west, central, south
- **Products:** coffee, espresso, tea, herbal tea

**Projection onto:**
- (Time, Products, *)
- (Time, *, *)
- (*, *, *)

**Dimensions:**
- All locations
- All products
- All time

**A two dimensional projection:**

**A one dimensional projection:**
particular projection in this lattice of data cubes: (a) which dimensions we currently consider relevant and (b) the appropriate level of detail for each relevant dimensional hierarchy.
level of detail identifies the cube in the lattice.

While identifying a specific projection in the data cube corresponds to specifying the desired data abstraction of the raw data, in multiscalar visualizations we need to specify both the data and visualizations. Figure 1 shows a simple lattice and projection.
The lattice of data cubes
the relevant dimensions identifies which projection of that cube is needed.
Visual Abstraction
1 Introduction

When exploring large datasets, analysts often work through a process of "overview first, zoom and filter, then detail on demand" (14). Multiscale visualizations are an effective technique for facilitating this process because they change the visual representation to present the data at different levels of abstraction as the user pans and zooms. At a high level, because a large amount of data needs to be displayed, it is highly abstracted. As the user zooms, the data density decreases and thus more detailed representations of individual data points can be shown.

The two types of abstraction performed in these multiscale visualizations are data abstraction and visual abstraction. Data abstraction (e.g., aggregation or selection) change the underlying data before mapping them to visual representations. Visual abstractions change the visual representation of data points (but not the underlying data itself) to provide more information as the user zooms.
2 Related Work

In this section, we review several existing multiscale visualization systems, focusing on how the systems perform both data and visual abstraction. Data abstraction refers to transformations applied to the data before being visually mapped, including aggregation, binning, sampling, or statistical summarization.

Visual abstraction refers to abstractions that change the visual representation (e.g., a circle at an overview level versus a text string at a detailed level), change how data is encoded in the retinal attributes of the glyphs (e.g., encoding data in the size and color of a glyph only in detailed views), or apply transformations to the set of visual representations (e.g., combining glyphs that overlap).

Multiscale Visualization in Cartography

Cartography is the source of many early examples of multiscale visualizations. Cartographic generalization [10] refers to the process of generating small scale maps by simplifying and abstracting maps such as removing details and reducing the complexity of lines, edges, etc.
3.2 Visual Abstraction: Polaris

Previously, we presented the Polaris database exploration tool consisting of three parts: (1) a formal specification language for describing table-based visualizations, (2) a user interface for automatically generating instances of these specifications, and (3) a method for automatically generating the necessary database queries to retrieve the data to be visualized by a specification. We later extended all three parts to support hierarchically structured data cubes [16].

In this paper, we only use the specification language from the previous papers. We use this language to describe a module within the menu graph identifying a multilevel visualization. In this section, we briefly review the components of a Polaris specification and introduce a graphical notation that succinctly captures the data and visual abstractions in table-based visualizations of hierarchically structured data.

A Polaris specification uses a formal table algebra to specify the table configurations of the visualization. Each expression is in the table.
### Visual Abstraction in Polaris

Each layer has three encodings.

- **Blank**: means no encoding allowed.
- **An empty slot**: indicates an optional data encoding.
- **A slot containing a field type**: indicates a required data encoding.
- **A primitive with no slot**: indicates a fixed value encoding.

#### Primitives:

- `abc` = text
- Bowler = point
- Arrow = line
- Pentagon = polygon
- Star = text or point

#### Color:

- `()` = ordinal palette
- `[ ]` = quantitative ramp

#### Size:

- `|` = height
- `--` = width
- `\` = both
3.2 Visual Abstraction

Previously, we presented the Polaris database exploration tool consisting of three parts: (1) a formal specification language for describing table-based visualizations, (2) a user interface for automatically generating instances of these specifications, and (3) a method for automatically generating the necessary database queries to retrieve the data to be visualized by a specification.
This paper presents: (1) a formalism for describing multiscale visualizations of data cubes with both data and visual abstraction, and (2) a method for independently zooming along one or more dimensions by traversing a zoom graph with nodes at different levels of detail. As an example of how to design multiscale visualizations using our system, we describe four design patterns using our formalism. These design patterns show the effectiveness of multiscale visualization of general relational databases.
3.2 Visual Abstraction: Polaris

Previously, we presented the Polaris database exploration tool ([15]), consisting of three parts: (1) a formal specification language for describing table-based visualizations, (2) a user interface for automatically generating instances of these specifications, and (3) a method for automatically generating the necessary database queries to retrieve the data to be visualized by a specification.

We later extended all three parts to support hierarchically structured data cubes.

In this paper, we only use the specification language from the previous papers. We use this language to describe a node within the query graph identifying a multivariate visualization. In this section, we briefly review the components of a Polaris specification and introduce a graphical notation that succinctly captures the data and visual abstractions in table-based visualizations of hierarchically structured data.

A Polaris specification uses a formal table algebra to specify the table configuration of the visualization. Each expression in the table
Abstract

Most analysts start with an overview of the data before gradually refining their view to be more focused and detailed. Multiscale pan and zoom systems are effective because they directly support this approach. However, generating abstract overviews of large data sets is difficult, and most systems take advantage of only one type of abstraction: visual abstraction. Furthermore, these existing systems limit the analyst to a single zooming path on their data and thus a single set of abstract views.

This paper presents:

1. A formalism for describing multiscale visualizations of data values with both data and visual abstraction.

2. A method for independently zooming along one or more dimensions by traversing a zoom graph with nodes at different levels of detail.
3 Multiscale Visualizations

In this section, we present our system for describing multiscale visualizations that support multiple zoom paths and both data and visual abstraction.

Rather than considering multiscale visualizations as a series of linear zooms, we think of multiscale visualizations as a graph, where each node corresponds to a particular set of data and visual abstractions and each edge is a zoom. Zooming in a multiscale visualization is equivalent to traversing this graph. Each node in this graph can be described using a Peloton specification that describes the visual representation and abstraction and can be mapped to a unique projection of the data cube, which is a data abstraction of the underlying relational data.

In the remainder of this section, we first review the two technologies we use to perform data abstraction (data cubes) and visual abstraction (Peloton). When reviewing Peloton, we also introduce a graphical notation for describing the key elements of a specification. Finally, we present how we can create a zoom graph of...
we think of multiscale visualizations as a graph, where each node corresponds to a particular set of data and visual abstractions and each edge is a zoom.

Zooming in a multiscale visualization is equivalent to traversing this graph. Each node in this graph can be described using a Peloton specification that describes the visual representation and abstraction and can be mapped to a unique projection of the data cube, which is a data abstraction of the underlying relational data.

In the remainder of this section, we first review the two technologies we use to perform data abstraction (data cubes) and visual abstraction (Peloton). When reviewing Peloton, we also introduce a graphical notation for describing the key elements of a specification. Finally, we present how we can create a zoom graph of...
3 Multiscale Visualizations

In this section, we present our system for describing multiscale visualizations that support multiple zoom paths and both data and visual abstraction. Rather than considering multiscale visualizations as simply a series of linear zooms, we think of multiscale visualizations as a graph, where each node corresponds to a particular set of data and visual abstractions and each edge is a zoom.

Zooming is traversing this graph.

In the remainder of this section, we first review the two technologies we use to perform data abstraction (data cubes) and visual abstraction (Polaris). When reviewing Polaris, we also introduce a graphical notation for describing the key elements of a specification. Finally, we present how we can create a zoom graph of the system.
3 Multiscale Visualizations

In this section, we present our system for describing multiscale visualizations that support multiple access paths and both data and visual abstraction. Rather than considering multiscale visualizations as simply a series of linear scenes, we think of multiscale visualizations as a graph, where each node corresponds to a particular set of data and visual abstractions and each edge is a scene. Zooming in a multiscale visualization is equivalent to traversing the graph.

Each node in this graph can be described using a Polaris specification that identifies the visual representation and abstraction and can be mapped to a unique projection of the data cube, which is a data abstraction of the underlying relational data.
Traversing the Zoom Graph
The End.