12-2014

Interactive Feature Selection and Visualization for Large Observational Data

Jingyuan Wang
University of Tennessee - Knoxville, jingyuan@utk.edu

To the Graduate Council:

I am submitting herewith a dissertation written by Jingyuan Wang entitled "Interactive Feature Selection and Visualization for Large Observational Data." I have examined the final electronic copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Computer Engineering.

Jian Huang, Major Professor

We have read this dissertation and recommend its acceptance:

Joshua Fu, James Plank, Michael Berry

Accepted for the Council:

Carolyn R. Hodges

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

Interactive Feature Selection and Visualization for Large Observational Data

A Dissertation Presented for the

Doctor of Philosophy

Degree

The University of Tennessee, Knoxville

Jingyuan Wang

December 2014

For my grandmother

Acknowledgements

I would like to express my gratitude to all faculty, staff, and students at Department of Electrical Engineering and Computer Science in University of Tennessee for the over five years' memorable experience during my stay in the Ph.D. program.

I would like to thank my Ph.D. advisor, Dr. Jian Huang, for providing an engaging atmosphere to do research, helping fund my graduate studies, challenging me to be a critical thinker and better colleague, and teaching me many lessons both in life and research that I shall never forget. I would also like to thank the members of my Ph.D. committee, Dr. Michael W. Berry, Dr. James S. Plank and Dr. Joshua S. Fu, for their intellectual contribution, guidance and feedback.

In addition, I would like to thank many people that are highly influential to my pursuits of knowledge along the way, Dr. James Ahrens and Dr. Jon Woodering at Los Alamos National Laboratory and Dr. Thomas Peterka at Argonne National Laboratory, my mentors of two summer internships who further motivated me and extended my research interests; Dr. Forrest Hoffman, Dr. Richard Mills and Dr. Yilu Liu, my colleagues in domain area who helped shape my dissertation work with their domain expertise; Wesley Kendall, Rob Sisneros, and Kimberly Shook, current and previous members of SeeLab and contributors to this dissertation work.

Last but not least, my family and friends have always been very supportive of me. I am very thankful to them.

Abstract

Data can create enormous values in both scientific and industrial fields, especially for access to new knowledge and inspiration of innovation. As the massive increases in computing power, data storage capacity, as well as capability of data generation and collection, the scientific research communities are confronting with a transformation of exploiting the advanced uses of the large-scale, complex, and high-resolution data sets in situation awareness and decision-making projects. To comprehensively analyze the big data problems requires the analyses aiming at various aspects which involves of effective selections of static and time-varying feature patterns that fulfills the interests of domain users. To fully utilize the benefits of the ever-growing size of data and computing power in real applications, we proposed a general feature analysis pipeline and an integrated system that is general, scalable, and reliable for interactive feature selection and visualization of large observational data for situation awareness.

The great challenge tackled in this dissertation was about how to effectively identify and select meaningful features in a complex feature space. Our research efforts mainly included three aspects:

1. Enable domain users to better define their interests of analysis;
2. Accelerate the process of feature selection;
3. Comprehensively present the intermediate and final analysis results in a visualized way.

For static feature selection, we developed a series of quantitative metrics that related the user interest with the spatio-temporal characteristics of features. For time-varying feature selection, we proposed the concept of generalized feature set and used a generalized time-varying feature to describe the selection interest. Additionally, we provided a scalable system framework that manages both data processing and interactive visualization, and effectively exploits the computation and analysis resources. The methods and the system design together actualized interactive feature selections from two representative large observational data sets with large spatial and temporal resolutions respectively. The final results supported the endeavors in applications of big data analysis regarding combining the statistical methods with high performance computing techniques to visualize real events interactively.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

With the advent of ever-growing data collection and supercomputing power, we are exposed to the ubiquity of massive data sets. Current computing power has greatly accelerated both simulation capabilities and the collection of experimental and observational data. Data sets with an increasing number of variables paired with greater spatial and temporal resolutions become the common subject for data analysis. It is crucial for domain scientists to differentiate and extract important information from such a complex problem space.

The common practice for domain scientists is to extract the large multi-variate data set into a reduced yet representative set of features through specific functions of mapping. With the increased size of data and growing computing capability, extracting features with much finer details is more affordable than ever before. While more features potentially contain more information, the amount of extracted features has become overwhelming to users - simple enumeration through these features is no longer plausible for analyzing the features in most cases. Interactive feature selection is called for such that a user can navigate, evaluate, and separate a complex problem space based on application-specific interest and significance.

Our end goal in this dissertation is to develop an effective, reliable and interactive feature selection system that facilitates the data analysis applications of situation

awareness and decision-making on large multivariate spatio-temporal observational data. Situation awareness has become one of the most popular focuses for today's data analysis applications. According to Endsley's model (Endsley, 1995), situation awareness includes three levels of hierarchical phases:

1. Perception of the elements in the environment;
2. Comprehension of the current situation;
3. Projection of the future status.

The model generally describes most of the objectives of the data analysis tasks for situation awareness. For each of the three phases, feature analysis could play a significant role especially for situation awareness in complex systems, such as the earth and the continental-scale power grid. In feature analysis, static features show different system conditions while temporal features reveal the transition of system conditions over time. The goal of data analysis for situation awareness is to identify the feature patterns that could lead to a deeper understanding of feature characteristics which deserve human attention, and responsive action by the decision makers in possible desired/undesired situation. With the ability to select significant features especially those previously unknown would allow the domain scientists to effectively grasp the central actionable information and carry out further research or responsive decisions.

While the process and the analysis purpose are straightforward, more research questions remain unanswered:

1. How to assist domain users to better define their analysis interest?
2. How to accelerate the process of feature selection?
3. How to present the intermediate and final analysis results in a visualized way?

Firstly, the selection of features is an application-specific problem. While most of the feature selection methods developed in the past focused more on selecting

features as the input for a better classification result, With respect to feature analysis for situation awareness, we suggested that feature selection should center around the user interest and let users to only focus on the features that deserve for further exploration. Therefore, this dissertation work proposed to quantify such feature selection interest and provide domain users a convenient way to differentiate the features using ranked numerical measurements. Two different quantification strategies are proposed in this work respectively for static and time-varying features. For static features, we developed three general quantitative metrics by evaluating the spatio-temporal properties of the features. The metrics are then used either alone of together to assist users to determine the subset of features for further exploration. For time-varying features, we evaluate the features in a more user-driven manner. By creating a time-varying feature template, users define their selection interest based on their possessed knowledge. And all the candidate features in selection are evaluated through a comparison with the template feature.

Secondly, similar to most of the other visualization applications, the performance of interactive feature selection is largely limited by the I/O performance as well as the large size of the data compared with the computing resources. The cutting-edge supercomputing architectures have enabled a great number of possibilities of extremely large scale data analysis and visualization. However, for most of the domain users and decision makers, supercomputers are rarely or limited accessible and the common computation environment is one of multiple standard commodity workstations. To achieve an interactive selection of features in a large data set on such computation environment, we have employed a series of high-performance computing techniques and algorithmic designs. In particular, we have proposed a simple but novel concept, generalized feature set. In addition to a multi-scale strategy, the concept of generalized feature set greatly simplifies the search space of feature selection and allows the users to analyze the features at different granularities with different level of information details.

Thirdly, in recent years, high-resolution tiled displays have become increasing popular for decision makers and domain scientists as the large-screens with a high pixel count could facilitate content-rich and simultaneous display of visual contents such as images, videos, data renderings, webpages and etc. As a result, an analysis environment with such tiled displays becomes a typical setting for large data analysis tasks. However, the use of the tiled displays still centers around displaying large-resolution visual contents that is not previously applicable on a single-monitor desktop setting. In this work, we proposed to divide a complete multi-viewport visual analytics application into multiple standalone visualization applications with specific functionalities. The design enables a much more flexible experience of visualization and data analysis on tiled displays. These standalone visualization applications could be combined in different ways for different analysis tasks. In addition, these visualization applications can be implemented for and deployed on different visualization platforms.

Overall, in this dissertation work, we have developed two scalable systems for interactive selection of static and time-varying features respectively in large data. The dissertation work are relevant to work from a number of different fields. In the following chapters, we first discuss about the related work by other researchers and describe and summarize the novel concepts and designs proposed in the followed chapters.

Chapter 2

Background

A general feature analysis system for spatio-temporal data sets consists of many components with which many research problems should to be concerned. In this section, I am going to describe the data sets to be used for the driving applications in this work and then review the state-of-art in the related fields as well as discuss about the their difference from and relationship with this work.

2.1 Applications Data Sets

2.1.1 Satellite Remote Sensing on Vegetation

Understanding and safeguarding the health of the planet's ecosystems is pivotal to our security, economical prosperity, quality of life, and the stewardship of our natural and cultural heritage. To this end, a key aspect is to understand and separate normal or healthy patterns of variation in an ecosystem from those that are abnormal and indicate threats to ecosystem health that may require intervention. Here we explore such spatio-temporal variations in forest ecosystems using remotely sensed vegetation patterns of growth and deterioration, or phenology.

The data we use consists of Normalized Difference Vegetation Index (NDVI) values, from the NASA MODIS (Moderate Resolution Imaging Spectroradiometer*). The whole data set covers the conterminous United States every 8 days at 250 meter resolution from 2000 to 2011. The overall spatial resolution of the data set is 19968 × 13568 making total size of the data amounts to 540GB. The MODIS instruments are launched by NASA in 1999 with Terra (EOS AM, N → S) and 2002 with Aqua (EOS PM, S → N). These instruments are designed to view the entire Earth's surface every 1 to 2 days and acquire data in 36 spectral bands, or groups of wavelengths. They offer an unprecedented look at terrestrial, atmospheric, and ocean phenomenology for a wide and diverse community of users throughout the world. The MOD 13 product provides Gridded Vegetation Indices (NDVI and EVI) to characterize vegetated surfaces. NDVI exploits the strong differences in plant reflectance between red and near-infrared wavelengths to provide a measure of "greenness" from remote sensing measurements (Justice et al., 1998; Kumar et al., 2011a).

The NDVI data product is also used widely for analysis in various domain researches. Wan et al. (2004) combine the NDVI and MODIS Land Surface Temperature and proposed a near-real time drought monitoring approach. Cleland et al. (2007) focused on study the planting phenology shifting and its relation to the global enviromental changes.

2.1.2 Wide Area Measurement of Power System

The power grid is a critical fixture in our current industrial era. Our society depends on its consistent availability. Power grid failures could paralyze a city, region, or in the worst case, an entire country. Situation awareness visualization plays a significant role in helping grid operators to better monitor the current environment and to recognize, prevent or recover from major system failures.

*http://modis.gsfc.nasa.gov/

Accurate dynamic wide-area measured frequency is highly desirable, especially as blackout events are becoming increasingly severe in power systems around the world. Power engineers have worked for decades to develop measurement tools for observing a power system's dynamics. The concept of building an Internet-based real-time GPS-synchronized wide-area frequency monitoring network (FNET) was proposed in 2000 by Qiu et al. (2001), and the concept was realized in 2005 (Zhong et al., 2005). The FNET system consists of two major components: a) frequency disturbance recorders (FDRs), which perform local GPS-synchronized frequency measurements and send data to a server through the Internet, and b) the information management system (IMS), which includes data collection and storage service, data communication service, database operation service, and web service. The measured wide area frequency information from FNET can be used to perform control and protection such as load shedding and emergency area separation. Up-to-date, the system has grown into a mature, multi-functional, low-cost phasor measurement system with stable performance and high accuracy. (Qiu et al., 2001; Zhong et al., 2005)

In this dissertation work, we have used two collections of FNET data set. The first one contains historical power grid measurement data collected from 49 FNET devices deployed over different locations within the East Interconnect on April 27, 2011. The second one contains the same kind of measurement data from January 2011 to September 2011 collected by 91 FNET devices in several major interconnects. The data set contains missing records in some of the locations or time steps. All the FNET sensors collect multivariate data (frequency, voltage and phase angle) at 10 times per second resulting in 864,000 time steps per day.

2.2 Computation and Analysis Environment

The system proposed in this work mainly targets to be used for data analysis and visualization on standard commodity multi-core workstations, which are readily available for most domain users. The computing machines of such type are normally

equipped with relative powerful parallel computing capability with multi-core CPUs and a mid-sized memory that fits a good amount of processing data. However, a one-time preprocessing on supercomputers might be necessary for extremely large data that requires much more computing resources.

The computing and analysis resources I used include:

- **Sonoran**: An SMP linux workstation with 8 Intel Xeon processing cores clocked at 2.1 GHz, 12 GB memory and an NVIDIA GTX480 graphical processing unit containing 448 CUDA cores and 1280 MB memory. It has a serial Linux file system with two 1.5 TB Seagate Barracuda 7200.11 SATA hard drives each of which has a 300MB transfer rate.

- **Tanami**: An SMP linux workstation with 12 Intel Xeon processing cores clocked at 2.4 GHz, 48 GB memory, two AMD FirePro V7900 Graphics Cards containing 1280 streaming processors and 2GB memory. It has a serial Linux file system with four 1.5 TB Seagate Barracuda 7200.11 SATA hard drives. The workstation is also equipped with 6 tiled-displays with 2 desktop monitors that form a 5760×3600 screen resolution.

- **Jaguar** [†]: A Cray XT5 machine located at the Oak Ridge National Laboratory. Jaguar consists of 298,592 2.2 GHz AMD Opteron processor cores with 598 TB of main memory and over 10 PB storage. The parallel file system on Jaguar is the Lustre file system [‡].

- **Nautilus**: An SGI Altix UV system consisting of one UV1000, 4 UV10s and 3 login nodes. The UV1000 has 1024 Intel Nehalem EX processing cores clocked at 2.0 GHz, 4 TB memory, and 8 GPUs. In addition, each UV10 provides 32 processing cores, 128 GB memory, and 2 GPUs. Nautilus has the Lustre file system with 1.3 PB capacity installed and mounted.

[†]Jaguar was used in part of this dissertation work, however, it is no longer available since it was upgraded to the newest Cray XK7 Titan supercomputer.

[‡]http://www.lustre.org

2.3 Related Works

In this dissertation work, we have developed a scalable system for interactive feature selection. The system was built upon components and functionalities highly related to several aspects. Here in this section, I will review the related works from different angles.

2.3.1 Feature Extraction

One of the challenges faced by the scientists in big data era is to derive previously unknown knowledge about the multivariate patterns from the large/complex data sets. Various of data mining techniques as well as interactive methods in variable space emerged with the purpose of extracting features. Example techniques include clustering (Kumar et al., 2011a; Tzeng and Ma, 2004; Woo et al., 2012), geo-spatial-temporal queries (Hadwiger et al., 2008) and variable space range queries (Maciejewski et al., 2012; Glatter et al., 2008). There are also various of techniques designed to extract special structures as features like vortices in vector field (example works are surveyed by Post et al. (2003)), however, our work mainly focuses on the general spatio-temporal features comprised of a subset of similar data elements.

Finding time-varying feature pattern, also referred as time-series subsequence, is one of the concentrated areas where data-mining community focuses to develop effective approaches and tools. It is also one of main analysis subjects in situation awareness tasks applications. Among the recent endeavors in this research field, approaches are developed to find a smaller number yet more important time-varying features. Liu et al. (2012) claimed to find minimum representative pattern sets that satisfies the three quality requirement: 1) produce a minimum number of representative patterns 2) restore the support of all patterns with error guarantee 3) and have good efficiency. With similar purpose, Yin et al. (2012) incorporate utility metric into sequential pattern mining, introducing the lexicographic quantitative sequence tree to extract the complete set of high utility sequences. Clustering

algorithm is also applied in mining temporal features, such as work by Möller-Levet et al. (2003). However, there is another different voice by Keogh and Lin (2005) who claimed that clustering of time-series subsequences is meaningless because clusters extracted from these time series are forced to obey certain constraints that are pathologically unlikely to be satisfied by any data set and therefore the clusters extracted by any clustering algorithm are essentially random.

2.3.2 Feature Selection

As the number of features generated by the feature extraction process becomes overwhelming for users to handle, feature selection techniques are also developed to filter and differentiate features.

Liu and Yu (2005) surveyed the popular approaches for feature selection in data mining field. They mostly fall into three categories: filter, wrapper and hybrid methods. The filter methods select features through specific evaluation criteria including distance, information, dependency and consistency. The wrapper methods select features by evaluating the quality of results of a mining algorithm with the candidate subset of features. Example works include work by Cantú-Paz et al. (2004) who used variations of filter methods as well as traditional wrapper approaches and work by Mitra et al. (2002) who proposed to achieve the effective feature selection by first partitioning the original feature set into a number of homogeneous subsets and selecting a representative feature from each of such clusters.

The related works share the similar concept with this work - trying to select the most important features. However, they mostly focus on searching a good subset of features to be used as input for classification while this work focuses on helping domain users to identify and analyze the useful but unknown features or feature groups instead. And also the interactive feature selection approaches proposed in this work are based on evaluation of their spatio-temporal characteristics instead feature qualities Wang et al. (2013). A more similar feature selection approach is proposed by

Le Moal et al. (2012). Instead of evaluating the features' spatio-temporal properties, their proposed method employs a feature relevance characterization procedure by evaluating the relevance between the possessed knowledge about the phenomena of interest with the extracted feature set and the most relevant features are then selected.

2.3.3 Feature Rendering and Analysis

With the spatio-temporal features extracted from the original data sets, effective visualization method is also essential for domain users to study the corresponding scientific problems. Visualizations of static features commonly employ transfer-function based techniques to render the regions of interest and provide the user with the ability of interactively selecting or changing the appearance of the features. Due to the complexity of ever-growing size of data, it has become prevalent to use multiple linked views to simultaneously show, explore, and analyze different aspects of multivariate data. Examples include the SimVis system by Doleisch et al. (2005) and the follow-up works. Query-driven visualization (Glatter et al., 2008) has also become a popular research topic. The main idea of query-driven visualization is to combine feature extraction process with the visualization process. The extraction of features are done through a compound boolean range query directly on the scalar data field based on users' specifications.

When it comes to temporal feature patterns, the visualization approaches are more designed using information visualization techniques to convey the information of temporal transition. There are many research works aimed at solving the problem of effectively visualizing temporal features. Weber et al. (2001) presented an approach for the visualization of time-series data based on spirals, different to classical bar charts and line graphs which are still largely used in domain fields. Pretorius and Van Wijkk (2006) presented a method for the visual analysis of multivariate state transition graphs. Their technique is based on attribute-based clustering which results in a significant reduction in complexity and serves as a useful analysis mechanism with

user-involved interaction. Lin et al. (2004) designed a system to interactively identify temporal feature pattern and visualize the whole time-series by transforming the time series into a symbolic representation, and encoding the data in a modified suffix tree in which the frequency and other properties of patterns are mapped onto colors and other visual properties. Kincaid (2010) proposed a Focus+Context approach to provide a means of navigating to and inspecting low-level signal details in the context of the entire signal trace. Very recently, Sips et al. (2012) developed a multi-scale analysis approach that captures interesting patterns in environmental time series. Their method consists of an algorithm to compute statistical values for all possible time scales and starting positions and a matrix representation of potentially interesting patterns using the statistical values derived.

2.3.4 Parallel Processing for Feature Extraction

The continuing growth of data size and computing ability have enable both challenges and opportunities. The traditional algorithms that are not specifically designed for large-scale data are highly restricted in providing useful support for large-scale data analysis. The analysis of extreme large-scale data sets requires an efficient solution for feature extraction.

There are many research works aimed to handle feature extraction techniques for large-scale data. Examples of such works like clustering include the works by Kumar et al. (2011b); Bahmani et al. (2012); Patwary et al. (2012) and Ferreira Cordeiro et al. (2011). K-means clustering is one of the most popular clustering algorithms. Kumar et al. (2011b) had implemented the parallel version of K-means clustering algorithm. However, the K-means algorithm relays on a optimal initialization to obtain the best clustering results. The recently proposed k-means++ initialization algorithm achieves this but with a downside of its inherent sequential nature and Bahmani et al. (2012) show how to drastically reduce the number of passes needed to obtain, in parallel, a good initialization. Patwary et al. (2012) implemented a scalable

parallel DBSCAN algorithm using the disjoint-set data structure. Similarly, Ferreira Cordeiro et al. (2011) aimed to design the algorithm for parallel clustering and they took the popular MapReduce approach.

For time-varying features, effective methods to explore important information from a large number of time-series data are also high needed. Rakthanmanon et al. (2012) presented a work that shows searching subsequences of time series under Dynamic Time Warping is much quicker than the current state-of-the-art Euclidean distance search algorithms. Mansour et al. (2011) aimed to efficiently construct suffix tree for very long strings and can be used in analysis for very long time series data sets.

2.3.5 Data Analysis for Climate and Earth Science

Data analysis is also essential for climate and earth science research. There are many observational data being collected with the purpose for scientific discoveries for global ecosystem, especially for the early-warning and decison-making ones. Kumar et al. (2011a) investigated methods of geospatio-temporal data mining of multi-year land surface phenology data for the conterminous United States as part of an early warning system for detecting threats to forest ecosystems. Their work focused on the quantitative segmentation of the global climate into ecoregions, also know as climate regimes Hargrove and Hoffman (2004). Kumar et al. (2011b) focused on the development of a massively parallel multivariate geographical spatio-temporal clustering code for analysis of very large data sets for identifying regions of similar ecological and environmental conditions. Kendall et al. (2011b) analyzed the effects of the flow field from lower levels of the Northern Hemisphere to the lower levels of the Southern Hemisphere especially on CO_2 exchange in between.

2.3.6 Data Analysis for Power System

The power system community has been aimed to measure the wide-area power grid operation and collect as well as archive the historical data sets. A power system frequency monitoring network (FNET) was proposed and established in recent years. It serves the entire North American power grid through situational awareness techniques, such as real-time event alerts, accurate event location estimation, animated event visualization, and post event analysis. (Zhang et al., 2010; Singh et al., 2011) With large amount of the data collected, there actually have been not many advanced visual analytics tools designed for the analysis job. The mostly-used visualization techniques in the domain field are the line curves or two-dimensional heat map of single variables in the data sets. Examples of the visualization show cases in the domain fields include works by Messina et al. (2006). Analysis-wise, Markham et al. (2011) used several years' worth of frequency measurements obtained from FNET to examine the occurrence of frequency extrema in the Eastern and Western Interconnections with respect to time. Mei et al. (2008) presented a clustering-based dynamic event location method that uses wide-area generator rotor frequency measurements. Although I do not work with non-measurement data specifically, there are works by Wong et al. (2009) to visualize the physics of the power grids, which ultimately determines the condition and stability of the electricity infrastructure, using a graph-based techniques and Overbye et al. (2003) developed an interactive 3D visualization method of the power grid network of the operational and economic data.

2.3.7 Scalable and Fault-tolerant Visualization Systems

A scalable visualization system needs to manage the data and coordinate the different visual analysis components efficiently such that the analysis of the data could be conducted smoothly. Several popular scientific visualization packages as well some research works have implement their own scalable visualization system.

Scalable I/O Performance

Scalable I/O access is central to large-scale data analysis and visualization applications as I/O can place very expensive burden on today's data scientist and domain experts due to that fact that accessing data stored in disk drives is still very slow compared to the greatly improved computation processing capability in most computing architectures including the top supercomputers in the word. As a result, the limiting factor in the scalability of such applications has shifted from computation to I/O (Childs et al., 2010).

On most of the high performance computing architectures and small grouped clusters, parallel file systems are installed for handling file I/O work in parallel. Ross et al. (2010) presents and illustrates the characteristics, basic concepts, design and challenges of the existing parallel file system architectures. One of the main tasks of a parallel file system is organizing storage hardware into a single logical space. Data files of large size are typically split up into regions of contiguous data and these regions are then distributed across storage elements, so that many disk drives and network links may be used to access different regions of the file. Many parallel file systems also store the file information and necessary internal data as metadata to organize and coordinate the data access process. Popular parallel file systems include GPFS, Lustre, PVFS and etc .

Additional to the parallel file systems, many parallel I/O libraries are available for users in needs including MPI-IO, HDF5, Parallel-NetCDF and etc. Kendall et al. (2011a) suggest that the design of the I/O layer in these applications should center around a partitioning strategy rather than a file format. Their implementation BIL (Block I/O Layer) optimize the parallel I/O process with the focus on data organization to provide efficient mechanism from data level and fit the access pattern of scientific applications. The BIL library also provides a simple programming interface to users and support Raw, NetCDF and HDF formats. This dissertation work has employed the library to in our systematic approaches to address the I/O

performance issue on our target computing environment. Liu et al. (2013) proposed a hierarchical I/O scheduling solution for collective I/O to address the I/O performance degradation issue caused by the non-contiguous access pattern of many concurrent scientific applications by optimizing the shuffling cost that is largely ignored by the previous research.

Tiled Display

Li et al. (2005) developed software tools for high-resolution display walls to alleviate the current limitation on visualization resolution and single-user window system for collaborative visualization. Eilemann et al. (2009) and Doerr and Kuester (2011) both developed scalable visualization system that mainly focus on parallel rendering and their use on large screens or power walls. These systems are mainly designed to handle the rendering work systematically and in parallel. ParaView[§] , a popular data analysis and visualization application in academic field, is able to execute the visualization tasks including some data processing tasks in certain level of parallelism based on a server-client architecture.

Fault-tolerance

Fault-tolerance is also a highly important factor in designing the computer software system. Cumulvs, a computational steering environment developed at the Oak Ridge National Laboratory, provides a mechanism for fault-tolerance which aims at recovering from faults that may occur in the running application (Mulder et al., 1999; Kohl et al., 2006). A fully automated fault-tolerant system were developed by Kola et al. (2004) for distributed video processing and off-site replication. The system has a hierarchical fault-tolerance mechanism including persistent job queues and job progress logs. Crawl and Altintas (2008) proposed a provenance-based fault tolerance mechanism for scientific workflows.

[§]http://www.paraview.org

Chapter 3

Multivariate Static Feature Selection in Large Observational Data

Derived from the original real-valued data set, a **STATIC FEATURE** is a specific multivariate characteristic that can be defined flexibly by domain users in each analysis application of observational data sets. Abstracting the raw data into static features enables a more informative analysis avoiding the excessive, though accurate but sometimes irrelevant information from the real-valued raw data set. In practice, a set of abstracted features has a relatively limited size which is greatly smaller than the original number of data elements, and hence it is more manageable for the human cognition. Each feature can be labeled using a unique numerical index, and each multivariate data element from the original raw data set will correspondingly be assigned to a static feature. The definition of static features and the process to generate them vary from application to application. It could be a cluster resulted from K-means clustering algorithm, a specific type of summary statistics, for example, the quantized number of standard deviation away from the mean, or even a highly descriptive characteristics such as "comfortable" and "uncomfortable" of the daily weather and so on.

A number of data analysis applications commonly employ traditional techniques of feature extraction that involve large-scale multivariate spatio-temporal data sets. With the growth of computing power and data size, extraction of static features with numerous fine details is more affordable than ever before. While more features potentially contain more information, the amount of extracted features becomes overwhelming to users, that is, simple enumeration of these features may be no longer feasible for analyzing the features in a lot of cases. Interactive feature selection is called for such that a user can navigate, evaluate, and separate a complex problem space based on application-specific interest and significance.

This chapter discusses the general practice of abstracting the real-valued data set into multi-scale static features, it includes our efforts in solving this computation-expensive task. We also proposed three quantitative metrics to effectively assist users in interactive selection and exploration of significant static features. In the section of application results, we will discuss the utility and practicality of the approaches.

3.1 Scalable Abstraction of Multi-scale Static Features

Due to the large volume of the multivariate raw data set, the abstraction of the multivariate real-valued observations into the informative static features is usually a computation-expensive task. Despite being a one-time preprocessing step, a scalable implementation of such a task is still necessary and useful. Regarding different analysis applications, the abstraction is also varied based on different purposes of analysis, In this section, I will discuss one of the most commonly used feature abstraction methods, hierarchical clustering, in terms of our adaption of the implementation of algorithm for large-scale multivariate data sets in the parallel computing environment. The implementation successfully abstracts the data set into

multi-scale features with effectiveness and accuracy. Additional information about other data abstraction practices are discussed in the application results in Chapter 6.

3.1.1 A Parallel Hierarchical Clustering Implementation

Feature abstraction is an important process that requires high efficiency and accuracy, especially for the large-scale data sets. In the dissertation, we proposed a customized parallel hierarchical clustering algorithm to abstract the data set into static features at multiple scales, which helps users to analyze the problems at varying granularities. Here, the hierarchical clustering is implemented in a bottom-up fashion. Small grained clusters are merged together as long as the distance between cluster centroids is below a preset threshold. As hierarchical clustering progresses to coarse levels, the distance threshold increases linearly. Each cluster is a multivariate feature regardless of a specific level or scale.

A proposed data-level hierarchy structure to represent the data is as follows.

- **Element**, every single data point is considered to be an element.
- **Set**, all identical elements are bundled into a set. Here, "identical" means less than a very small but tunable threshold value given by users when computing the distance between each other.
- **Group**, similar sets, close enough for a larger distance threshold given by users, are merged to form a group. Several groups can also be merged to a larger group than before. Each of the groups in the final clustering result is a cluster that is also referred as static feature in this work.

The clustering algorithm originated from the disjoint set algorithm. Given a list of elements, sets, or groups, the algorithm would find out the near elements, sets, or groups within a certain threshold distance and merge them all into a new one.

To fully utilize the parallel processing ability of current computing architectures, parallelization of the clustering is necessary. The implementation is designed in

Algorithm 1 Hierarchical Clustering Algorithm

```
function PARA_HCLUSTER(elements, threshold_list)
    sets = create_sets(elements, threshold = 0.0)
                              ▷ only elements that are exactly the same are grouped
    groups = create_groups(sets, threshold_list[0])
    current_groups = groups
    for all threshold in threshold_list[1...N] do
        prev_groups = current_groups
        current_groups = refine_grouping(prev_groups, threshold)
        write_groups(current_groups)
    end for
end function
```

a parallel fashion. It uses a variety of parallelization techniques including an floating point Morton Order (Connor and Kumar, 2009) implementation of parallel sample sorting algorithm and the parallel I/O library, Block I/O Layer to boost the computation process.

Due to the size of the whole MODIS data set exceeding 500GB in size and the nature of the proposed algorithm that is designed based on in-memory computation, the hierarchical clustering process of the MODIS data that has to be performed on a supercomputer with more memory and computing resources. In practice, we usually consider the yearly phenological patten as the minimal research target. Each 46-element NDVI value tuple in the 11-year data is regarded as a 46-dimensional variable. Overall, the complete run of the hierarchical clustering includes a total of 2,980,184,064 46-dimensional floating data elements.

3.2 Quantitative Metrics for Spatio-temporal Feature Selection

Selecting meaningful features is the core in the analysis of scientific data. As multivariate scientific data sets are often large and complex, it is difficult to define general features of interests that are significant to scientific applications.

We assume that all features can potentially play an important role. Hence, several techniques directly render features in their original spatio-temporal level and the users have to determine what kind of features deserve further exploration. The assumption still needs to be improved for handling features that may be noise-corrupted, redundant, or less informative.

The purpose of developing metrics is to provide a general way of quantifying significance among a large number of features. Our CO_3 metrics, concentration, continuity, and co-occurrence, both spatially and temporally encapsulate properties that are readily identifiable in the physical space. The metrics represent three desirable properties when exploring interesting features by domain experts.

CO_3 metrics are defined on a per-feature basis and assume that the 4-dimensional space including the spatial and temporal domain has been partitioned into coarse grained bins, referred to as *regular bins*. All dimensions are treated equally in the partitioning. In general, the granularity of each regular bin is defined in the 4-dimensional space [x, y, z, t]. As for different focuses of data set analysis, MODIS is partitioned spatially to study the distribution of yearly vegetation growing pattern in geographical space, while FNET is partitioned temporally to study the dynamics of power grid over time. Examples of granularities can be [5 km, 5 km, −, 1 year] or [−, −, −, 1 second], where the '−' symbol denotes an undefined or unpartitioned dimension.

The CO_3 metrics are computed based on the distribution of static features in the partition of the physical space; hence, the choice of bin size affects the values of the metrics. The CO_3 Inspector System empirically provides a pre-set of bin sizes: 5, 10, 15 and 20 km for MODIS and 1, 2, 5, 10 second for FNET. These pre-set bin sizes are based on the rough spatial/temporal scale of application problems that domain experts are interested in. For instance, 10 seconds is considered to be a long period of time in which power transmission on the grid would vary greatly.

To properly define these metrics for feature properties, we need the following notations and quantities:

F_i: Static feature i

E_i: Number of elements of F_i.

E_{ib}: Number of elements of F_i in regular bin b.

t: The percentage threshold for identifying significant bins.

For a given static feature F_i, the set of **SIGNIFICANT BINS** is the smallest set needed to represent a percentage t of all data points belonging to static feature F_i. For example, in Figure 3.1, a static feature F_i contains 28 data points and is spread over 5 bins, A through E. We sort the bins in decreasing order of E_{ib} and then traverse the array computing the prefix sum of E_{ib}. We stop the traversal as soon as the prefix sum has reached t. For a value of $t = 90\%$, the significant bins would be A through D. The concept of significant bins elegantly handles noise-like anomaly data, the choice of t is application dependent.

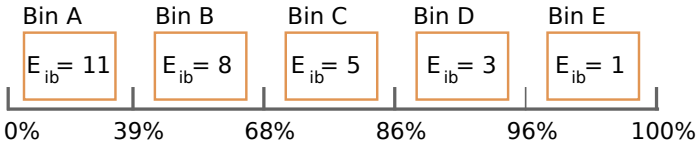


Figure 3.1: An illustration of determining significant bins. Given a static feature (F_i) and the number of its data points per bin (E_{ib}, in decreasing order), the set of significant bins is the smallest group of bins that can represent F_i's presence above a given percentage threshold (t).

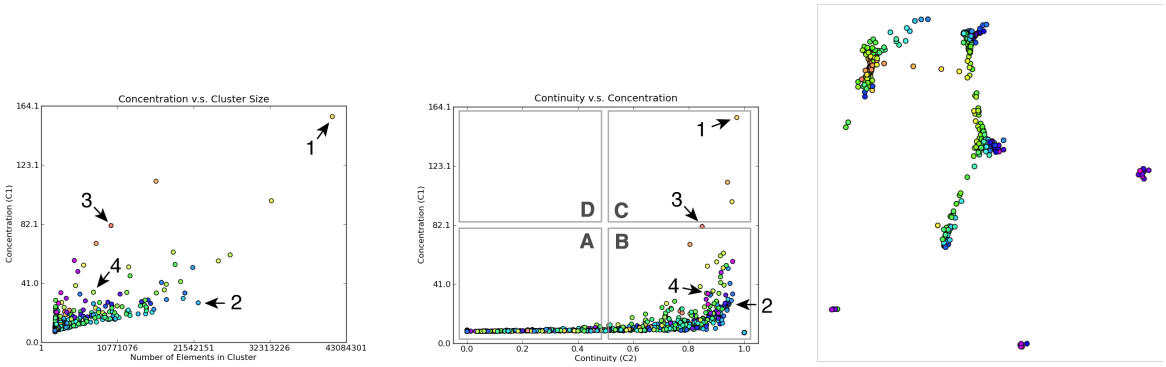
In the following subsections, we describe the three CO_3 metrics.

3.2.1 Concentration

The concentration metric, C_1, denotes the average occupancy of bins in the set of significant bins for a given static feature. It indicates the properties of a static feature with respect to both physical distribution and size and is calculated as:

$$C_{1i} = \frac{E_i * \gamma}{K_i} \tag{3.1}$$

where γ is the percentage of elements within significant bins for a given static feature F_i.



(a) Concentration *vs.* Feature size **(b)** Concentration *vs.* Continuity **(c)** Co-occurrence graph layout

Figure 3.2: Statistics views based on CO_3 metrics. Various examples of utilizing CO_3 metrics in visualization and analysis. The utility of the visualizations in the sub-figures (along with corresponding labels) are discussed in Sections 3.2.1, 3.2.2, and 3.2.3. (Year 2003, 5 km bin size)

Since C_{1i} depends on K_i, the number of significant bins, this guarantees that C_1 is not affected by outliers of the data in the static feature. Highly concentrated features have a high representation in a small number of significant bins and therefore have a high C_1 value. Static features with a small representation across bins will stack on the lower end of the C_1 axis. A concentrated feature can be a dominant pattern across a large portion of the physical space because of its large volume of data elements. It can also be a smaller-sized feature representative of certain locale in the physical space.

Figure 3.2a illustrates the space formed by concentration *vs.* feature size. The metrics are computed using a 5 km bin size. On a 250-meter resolution grid, this amounts to 400 geographic locations in every bin. A C_1 value of 200 or more indicates that a static feature monopolizes more than half of its significant bins. When a user examines highly condensed patterns such as vegetation damage due to insect infestation, those feature patterns are small yet highly concentrated. The static features corresponding to them will not appear among the large ones. The search should start from the left side of Figure 3.2a, populated by smaller static features.

Also in Figure 3.2a, several individual static features are labeled for comparison. Static feature "1" and "2" are both large but have very different concentration properties. Static feature "1" is one of the largest features on the continental U.S., yet it is so concentrated that it takes up almost half of each physical bin. That static feature happens to correspond to the mountainous areas of the western United States. Static feature "2" is large but does not monopolize any 5 km-square geographical bins. Static feature "2" distributes over the middle and eastern part of the United States. Static feature "3" is similar in size to static feature "4", but is more concentrated with its spatial presence concentrated on lakes and other water bodies. Static features appearing at the right-bottom corner of this plot are likely widespread noise in the data.

3.2.2 Continuity

C_2 denotes the continuity of significant bins for a given static feature. Bins are connected if they comprise spatio-temporally continuous regions. Connected significant bins are grouped into *significant regions* and C_2 is calculated as:

$$C_{2i} = 1.0 - \frac{R_i}{K_i} \tag{3.2}$$

where R_i is the number of significant regions and K_i is the number of significant bins. Hence, C_2 can range from 0.0 (no bins connected) to 1.0 (all bins connected, 1.0 not included).

When paired, continuity and concentration create an interesting space. We believe the C_1 *vs.* C_2 space can be divided into four areas in which static features that fall in the same area share similar spatio-temporal properties. For example, Figure 3.2b shows a sample plot of C_1 *vs.* C_2 with labeled regions. In this space, low concentration and low continuity likely represent noisy data elements (A); high concentration and high continuity represent a static feature that is well represented in distinct spatial regions of the data (C); and low concentration and high continuity

could easily represent elements of data that define "normal" data elements for given regions (B). Defining features of interest is entirely dependent on the application however. Figure 3.3 provides a map view of the static features in regions (A), (B) and (C).

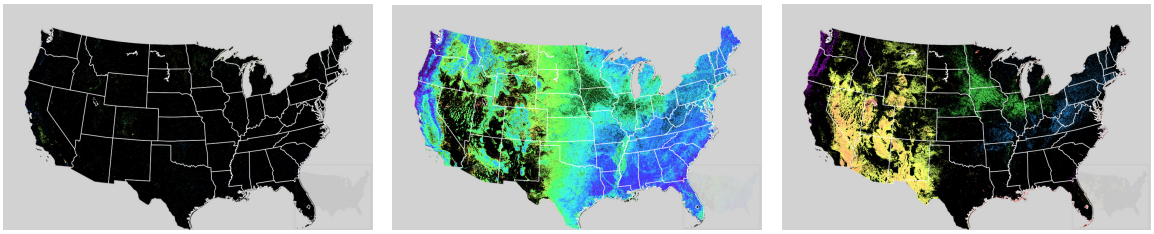


Figure 3.3: Static features in quadrants A, B and C (left to right) in Figure 3.2b.

Note that static features "1" and "3" in Figure 3.2a are still clearly distinguishable in Figure 3.2b. From that, we can tell both of those static features are highly concentrated and continuous and are likely features representative of a geographic area. Static feature "2" in Figure 3.2a is also highly continuous as the agricultural growing pattern represented by static feature "2" is more or less common in the middle and eastern US though not prevalent.

3.2.3 Co-occurrence

While concentration and continuity quantify global properties of a single static feature, we also desire to assess static features locally and within the context of one another. Co-occurrence, or C_3, measures the degree to which static features reside near each another (i.e. are collocated) and assists in the analysis of relationships between features. Unlike C_1 and C_2, C_3 is calculated from all bins, not just significant bins.

$$C_{3ij} = \frac{\sum_{b \in V_{ij}} min(E_{ib}, E_{jb})}{(E_i + E_j)/2} \tag{3.3}$$

For two static features F_i and F_j, V_{ij} is the set of regular bins in which F_i and F_j overlap. C_{3ij} measures how much F_i overlaps F_j in spatial presence on the granularity of spatial bins. Hence, C_3 will range from 0.0 (no overlap) to 1.0 (perfect spatial overlap). This metric is very well-conditioned to be directly used for edge weights in a force-directed graph layout algorithm. We threshold edge weights and filter out edges before performing the graph layout. Figure 3.2c is an example with a threshold corresponding to keeping only top 40% of edges and a bin granularity of 5 km. We omitted edges to reduce over-plotting.

With concentration and continuity, users can specify significant features based on the strong or weak properties; however, co-occurrence is more complex to understand because co-occurrence can not be examined using the concept of 'high' or 'low'. However, by employing a graph layout algorithm to embed the features into a two-dimensional graph, users can better visualize and analyze this metric. In the graph, the position of a particular feature has no physical meaning. The distances between features are the only measurement related to C_3. If features are close to each other in the graph, it means these features are near each other spatially or they occur in similar period of time.

The significance of C_3 is shown by our driving applications. Climate scientists are always interested in discovering exact causes of abnormal growing patterns. Two co-occurred features imply certain ecological scenarios. It could be that they are both consequences of the same event, like unexpected regional drought. Or it could be that one of them is the cause of other co-occurred features. Similarly for the power grid application, unusual events that occur shortly before or after abnormal power grid operation states, like large-scale frequency oscillation, are significant. Understanding the reasons for and the consequences of an abnormal event is crucial for handling similar occurrences in the future.

Although initially more complex to understand, C_3 actually presents a great deal of information about feature combinations which is often neglected or missed in traditional attribute analyses. In Chapter 6, we present some interesting groups

of features discovered from the co-occurrence graph of the CO_3 Inspector system that were not known a priori.

Chapter 4

Multivariate Time-varying Feature Selection in Large Observational Data

As introduced in Section 3.1, static feature is a general concept that can be interpreted as a specific multivariate characteristic which is defined by users ahead of the analysis applications. No matter how the static feature is defined, each static feature can be identified by a discrete and unique numerical index in processing. The interpretation of the static features comes after any data processing steps related to the static features.

Time-varying Feature is defined as a sequential list of static features that are consecutive in the time span, and it describes a time-varying characteristic pattern in the feature space. The time-varying features can be interpreted as temporal events and categorized into different semantic groups if necessary. For example, a time-varying feature can be a set of sequential humidity conditions that start from drought and gradually recovered to a normal humid condition at a particular geo-spatial location, and it could also be a time-varying feature describing the hurricane intensification process varied from regular windy to different hurricane intensity scale. For these two examples, both the humidity conditions and the hurricane intensity scale are the static features that provide the time-varying features.

In situation awareness and decision-making applications, time-varying information is highly essential in order to assess the current situation and estimate the future possibility based on the historical records. Analyst or decision-maker usually realizes that it is useful to relate certain target temporal events with similar historical trends compared with many others. The corresponding distribution and magnitude in both spatial and temporal domain not only can deepen the domain users' understanding of particular time-varying features but also provide a guidance for further analysis. The whole process of analysis generally follows the same style in this dissertation, which starts from basic or incomplete knowledge about time-dependent events, and then users are guided by using the metrics or tools to discover extra details from the iteratively try-and-refine analysis.

For time-varying feature analysis, selection of significant time-varying features is also highly essential and valuable but difficult for large spatio-temporal observational data sets.

However, the analysis tasks for large scale data sets become highly challenging as the problem space grows dramatically. Defining the significance of time-varying features is necessary, but there are no specific design metric developed to assess the significance heretofore. It is proposed in this dissertation to connect the user interests to the time-varying significances which are straightforward as long as the user interests are well defined. The following sections articulate such process and challenges.

4.1 Generalized Feature Set

In a structured spatio-temporal observational data set, a time-dependent sequence can start from anywhere with varied length and in any arbitrary spatial location. As a result, the number of time-dependent sequences is extremely large and very unlikely to be operational by a traditional click-and-view analysis process even just for a relatively small data set.

Similar to our proposal of the CO_3 metrics in the Section 3.2, which are developed to define the significance of the static features, the significances for time-varying features are determined quantitatively as well. However, different from the CO_3 metrics, which are developed based on the spatio-temporal correlation among the static features and pre-generated automatically using their mathematical definitions, the significances for time-varying features are determined on the fly and rely on the user's interests and inputs. As the basis for the time-varying feature selection, we proposed a simple but novel concept, **Generalized Feature Set**, which includes a group of generalized features that fall into the following two categories:

- **Generalized Static Feature** is a set of static features with their percentage information. The static features are sorted in a decreasing order of percentage and do not necessarily sum up to 100% since a few the less occurred static features are discarded in some cases.

- **Generalized Time-varying Feature** is a sequential list of generalized static features. Each static feature represents a time step of equal length.

Generalized time-varying features are used to represent temporal feature patterns with uncertainty, and each time step contains a number of static features of different and sorted occurrence possibilities. Generalized static features and time-varying features are widely used in part of this dissertation work in terms of the selection of multivariate time-varying features. The multivariate raw data set comprised of real values is firstly abstracted to multi-scale static features and then generalized to the format of generalized static features, which keep the multiple level of incremental granularities unchanged. In this case, a generalized static feature at the coarsest level is a single regular static feature with a 100% occurrence weight. At a relatively fine granularity level, the generalized static features normally contains more than one static features. The static features with high occurrence possibilities produce more effects in the processing of selecting significant time-varying features than the features with low possibilities produce.

In the process of generalization from regular static features to generalized static features, the structured grid of the regular static features is partitioned into a relatively small block so that the different levels of generalized states make a clean tree-typed data hierarchy. For example, a block dimension of $[256, 256, 1, 1]$ is a practical example that generalizes the static features only at the dimension of x and y spatially. An additional parameter of number of maximum static features (*depth*) can be specified to discard parts of the static features that are less occurred.

The further analyses are all based on the generalized static and time-varying features. Additionally, generalized time-varying feature is also used to describe the user-defined time-varying feature patterns and prove to be straightforward and flexible.

4.2 Time-varying Feature Matching

4.2.1 Similarity between Generalized Static Features

In contrast to the similarity between two single-variable real-valued data elements which can be computed directly from the absolute difference of the two values, through euclidean distance or other numeric distance computation methods, the similarity of two generalized static features is more complicated. The comparison between two generalized static features are based on the comparison between their regular static features corresponding.

Suppose two generalized static feature P and Q both can be represented by their corresponding sorted regular static features $p_{(1...m)}$ and their weighted percentage $\omega_{(1...n)}$:

$$P: \quad [(p_1, \omega_1), (p_2, \omega_2), ..., (p_m, \omega_m)]$$
$$Q: \quad [(q_1, \omega_1), (q_2, \omega_2), ..., (q_n, \omega_n)]$$

As the features are discrete and the sequence can be re-ordered since it is easy to determine the feature distance, The 100% occurrence of both P and Q can be separately divided into four static feature sets:

F_{shared}	:	a set of static features shared in both P and Q, the weights are determined by the minimum of ω_p and ω_q.
$F_{unique_u,q}$	:	a set of unique features excluded F_{shared} from P and Q respectively.
$F_{shared\_u\_p,q}$	:	the shared uncertain static feature distribution.
$F_{unique\_u\_p,q}$	:	the unique uncertain static feature distribution.
$F_{uncertain\_p,q}$	:	a combine set of $F_{shared\_u\_p,q}$ and $F_{unique\_u\_p,q}$.

A better illustration is presented in Figure 4.1. The different strategies of handling of $F_{uncertain\_p,q}$ result to several different variations of similarity metrics between two generalized static features. Here is the list of potential proper metrics of distance from Q to P are:

$$D_1 = 1.0 - \frac{Min(F_{shared\_u\_p}, F_{shared\_u\_q})}{F_{shared} + F_{unique\_p} + F_{uncertain\_p}} \tag{4.1}$$

$$D_2 = 1.0 - \frac{F_{shared\_u\_p}}{F_{shared} + F_{unique\_p}} \tag{4.2}$$

$$D_3 = 1.0 - \frac{F_{shared\_u\_p}}{F_{shared} + F_{unique\_p} + F_{uncertain\_p}} \tag{4.3}$$

$$D_4 = 1.0 - \frac{Max(F_{shared\_u\_p}, F_{shared\_u\_q})}{F_{shared} + F_{unique\_p} + F_{uncertain\_p}} \tag{4.4}$$

All the four distance metrics are supported in the selection of time-varying features. However, the D_3 metrics are set to the default method that only exactly match static feature are considered to the same between two generalized static features.

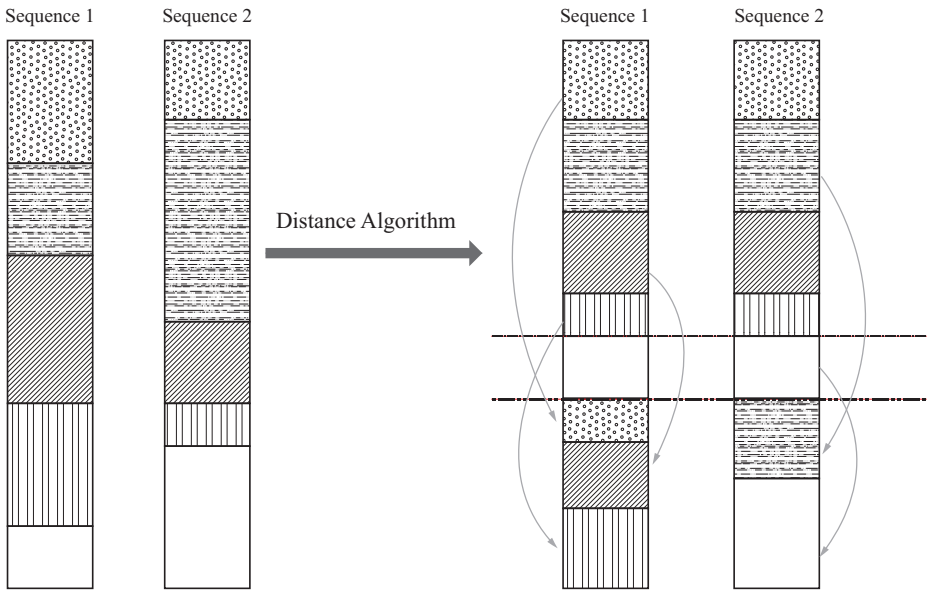


Figure 4.1: Distance metrics between generalized state features.

4.2.2 Similarity between Generalized Time-varying Features

In most application analyses on temporal feature patterns, two time-varying features are usually considered to be similar patterns as long as they follow the same variation pattern regardless of the temporal size of the time-varying features and the exact details about the appearance and duration of a certain feature. In other words, the exactly matching method of comparing two time-varying features which is normally done through scaling to the same size firstly in time dimension and then computing the accumulation of the pair-wise distances of the static features at each time step, is not the best choice. Instead, I apply the Dynamic Time Warping (DTW) algorithm to compute the similarity between two time-varying features here.

DTW is a well-known technique to find an optimal alignment between two given time sequences with certain restrictions. DTW algorithm is widely used and proves its usefulness in applications such as speech recognition and many time-related data mining applications. Müller (2007)

By utilizing DTW distance to determine the similarity between time-varying features, we assume that a time-varying feature that is aligned with another time-varying feature by less cost is considered to be similar to each other.

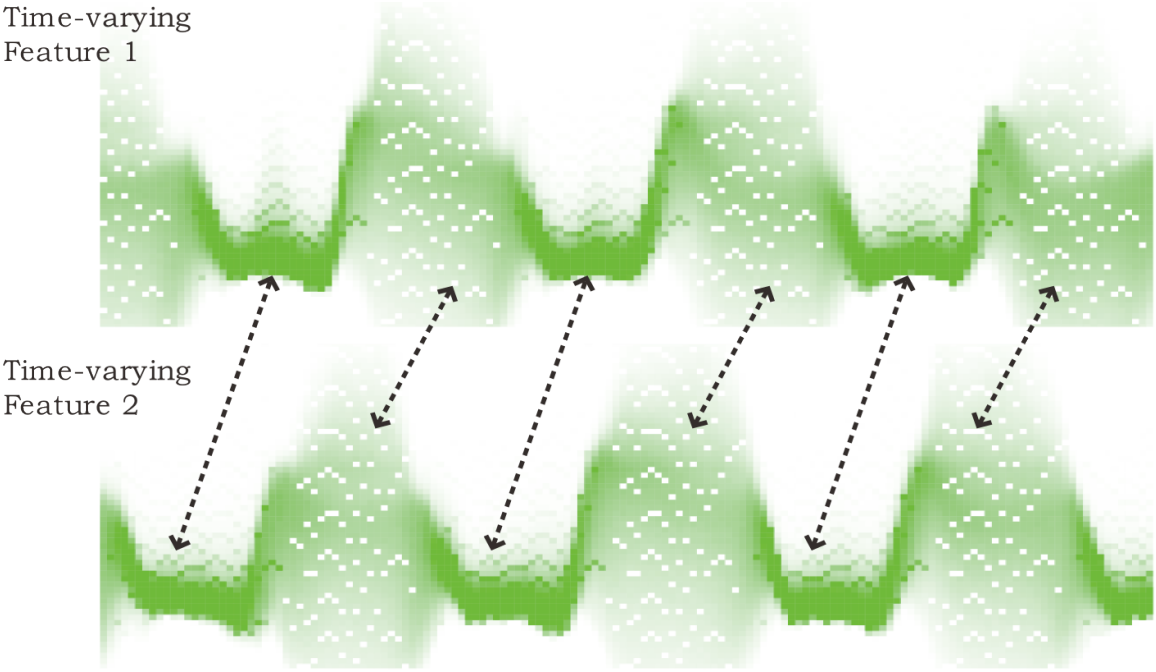


Figure 4.2: Example alignment of two time-varying features within the MODIS data set using DTW algorithm.

The classical dynamic algorithm and its variant methods can be implemented by using a dynamic programming method. The modified DTW variation are proposed to speed up the algorithm and to better the control of warping path at the same time. Keogh and Ratanamahatana (2004) However, the majority of such DTW variants are designed for the real-valued time sequence matching. It is not applicable for the matching between discrete static features which are not presumably or necessarily correlated with each other continuously. For example, a number of algorithms use a lower bounding distance measure to speed up the sequential scan search and prove to effectively improve the DTW alignment process by a fast pruning of the sequences that may not be the best matches. These lower bound measures work based on the assumption that the sequence elements are a single continuous real value such that a

lower bound can be then computed. However, with respect to the generalized time-varying features comparison, the lower bound concept does not be applied because the generalized static features represent a certain multivariate characteristic instead of a continuous value.

Pseudo-code for the classical DTW algorithm is as follows:

Algorithm 2 Distance between Time-varying Features

function DTW_DISTANCE(Time-varying Feature S (s_1, s_2, ..., s_n), Time-varying Feature T (t_1, t_2, ..., t_m))
    Initialize cost matrix DM with $n+1$ rows and $m+1$ columns
    for all $i \leftarrow 1$ to n **do**
        $DM[i, 0] \leftarrow \infty$
    end for
    for all $i \leftarrow 1$ to m **do**
        $DM[0, i] \leftarrow \infty$
    end for
    $DM[0, 0] \leftarrow 0$
    for all $i \leftarrow 1$ to n **do**
        for all $j \leftarrow 1$ to m **do**
            $cost \leftarrow$ Distance(s_i, s_j)
                ▷ Distance between two generalized static features
            $pcost \leftarrow$ Minimum($DM[i-1, j]$, $DM[i, j-1]$, $DM[i-1, j-1]$)
                ▷ Previous mininum total cost
            $DM[i, j] \leftarrow cost + pcost$
        end for
    end for
    return $DM[n, m]$
end function

4.3 General Time-varying Feature Selection Strategy

The selection of significant time-varying features relies on a scalable search for time-varying features which is very challenging in large spatio-temporal observational data sets. The approach proposed in this section utilizes the user-defined templates

to query significant temporal features. By allowing the users to determine the importance of features based on their domain expertise and knowledge, the approach can assist in selection of similar significant features from the whole data set. Considering the time-varying features as events, the spatial and temporal distribution and magnitude of such events are then to be visualized and analyzed simultaneously in coordinated views.

4.3.1 Defining Time-varying Features of Interest

Defining time-varying features of interest is the very first step to determine the significance of time-varying sequences in the data set. The interests of such time-varying features defined by the domain users are represented by using a generalized time-varying feature and serve as the search "keyword" for querying significant time-varying features. It is named as **Time-varying Feature Template**. If a time-varying feature and the feature template have a relative smaller DTW distance in matching, they are considered to be more important to the domain users than others. In a word, the significances of time-varying features are determined by the degree of similarity between their similarities with the template feature.

The generalization of static features and the use of the concepts of generalized static and time-varying features allow a creation of simplified time-varying feature template, a scalable data set structure and a fuzzy query that allows uncertainty in the feature selection.

Methods to create the time-varying feature search templates include:

1. **Generalized from examples at runtime**. This method typically takes an input of multiple and very often a mid-sized number of example time-varying features that are selected directly and interactively from the observational data set. The static features at each time step are aggregated together and generalized to a new generalized static feature with a number of regular static features sorted by the occurrence percentage. The static features paired with

a relative small percentage weight are discarded to the extent specified by the users. In addition, the occurrence statistics of all static features are collected in order to adjust the weights of the static features by PageRank algorithm, such that the static features that play more important roles in the temporal transition have higher impact on the matching between candidate and template time-varying feature.

2. **Sketched interactively in real-value variable space**.

   The sketch-based method to generate a time-varying feature template is quite flexible for domain users with not too much knowledge about the location or time of interested time-varying feature patterns but clear about the exact multivariate behavior. The domain experts can interactively sketch the time-varying features using a varied-sized and varied-transparent painting brush to express their target time-varying feature patterns. The sketch is then processed into static features in each time steps and then generalized into a generalized state. The process is called "flatten the sequence" in the interactive system described in Section 6.2.2 An extra input argument of the amount of regular static feature in each time step is also considered if an approximation with better query performance but relative small lose of details in the template is preferred.

The sketch-based method of creating time-varying feature template could be applied either directly or as an interactive refinement of an existed template such as one generated using the example-based method. A typical process of creating time-varying feature template is illustrated in Figure 4.3.

4.3.2 Multi-scale Searching of Time-varying Features

In order to select significant time-varying features, specifically to say, the similar time-varying features compared with our feature template in our case, we apply multi-scale

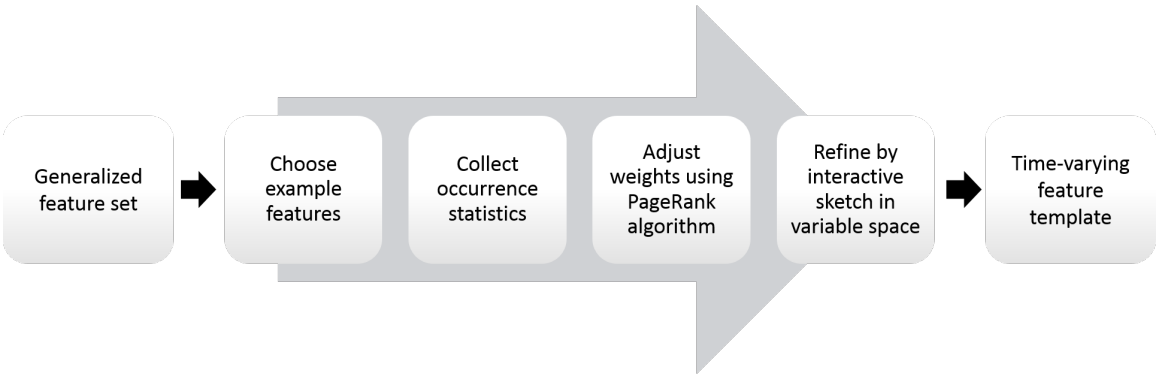


Figure 4.3: The process of creating a time-varying feature template.

strategy in search of similar time-varying features within large-scale spatio-temporal observational data set. In the general spatial and temporal space, a generalized static state can be expanded to a list of regular static states in neighboring spatial regions and periods of time. In a different way, the multi-scale feature sets can be identified as a tree data structure. Each tree node is a generalized state that summarizes all of its child nodes and the leaf node is a regular static state.

The multi-scale searching process, also referred as coarse-to-fine strategy in our application section, starts by looking for the better matched time-varying features among all the ones at the coarsest level of the feature sets and prune the non-related ones in the search of next level. The returned result refines the target portion of the feature search process in the next fine level of data set. The practice is very straightforward for time-varying feature sets with large spatial resolution but relatively small temporal resolution. However, for the feature sets that contains many time-varying features lasting for a extremely long period of time, like the FNET data set, the temporal domain is firstly partitioned into relatively small segments. Each of the segments is set to the same length with the longest expected time-varying features to be selected. For instance, a feature template of 100 time steps should not be compared with one of 10000 time steps. A proper setting would be 10 times longer and the number is tunable to users. Additionally, an overlapped ghost region

in temporal space is padded to the beginning and ending of each temporal partition in search to avoid missing selection across the segment boundaries.

The multi-scale strategy has actualized not only a better performed searching of time-varying feature with the user-defined template using DTW algorithm but also a robust alignment between the feature template and target features.

Chapter 5

Scalable ManyView Data Analysis and Visualization

In recent years, high-resolution displays have become increasingly common and important to decision makers and scientists because large-screens with a high pixel count facilitate content rich and simultaneous display of visual contents such as images, videos, data rendering, webpages and etc. In visual analytics tasks, tiled displays are also favorable due to its extended screen that provides sufficient space for different viewports with different analysis tasks.

This dissertation work has mainly targeted on the computation and analysis environment with one or more high performance multi-core workstations that equipped with such tiled displays making up a large resolution of display screen.

The systems, frameworks and architectures developed to better utilize such environment has mostly placed their focus on the scalability with respect to the size of the data or low-level rendering parallelization capability using the graphics hardware (Eilemann et al., 2009; Doerr and Kuester, 2011; Childs et al., 2010). These works provide great help to the data analysis and visualization field in their specific applications. However, for many of the other visual analytics applications, in which a very high-resolution volume rendering is not necessary, however coordinated views

to support task-parallel applications are more demanded. Hence, a new scalable, adaptive and interactive data analysis and visualization framework is largely needed.

The two key factors are how to make best use of the parallel computing resources including multi-core CPU and large in-core memory and how to make best use of the extended display screen. As a result, We have made the effort to develop such a systematic framework comprised of different approaches that are discussed below.

5.1 Client-Server Design

In most of the popular visualization application settings, the interactive user interface often comprises of multiple coordinated viewports, within each of which the users manipulate on a certain subset of the data sets or processed intermediate results being rendered. The rendering of the viewports are normally coordinated and updated upon user interaction in any of them. These viewports are very often implemented together in a standalone GUI window by a serial process which only allows for limited data processing work. The performances of such user interfaces are very limited when it comes to our targeted analysis environment. The whole interface is bottlenecked by the individual viewports.

The design scheme of the system is to separate the visualization and interaction work from the computation-expensive data processing work. In our design, the system has a servers-clients architecture. There are two types of servers - data server and state server, and multiple visualization clients. In this work, the whole feature analysis process is divided into several components based on different functionality and application characteristics. Each component is implemented as a visualization client and connected to both data server and state sever through socket.

5.1.1 State Server

In a regular multiple-viewport visualization user interface, the rendering, processing and coordination are achieved through either shared-memory communication, signal and callback functions. In the proposed system framework, visualization viewports are designed to run in separate processes such that they can be launch and terminate independently. The communication between the different visualization viewports which are named Visualization Client are achieved using State Server, a serial state machine that maintains all necessary communication messages, also called representational states, such as time step, interactive selection, rendering transfer function and etc.

The state server is designed with a clean data structure, a single red-black tree. Any processing on the communicational states are initiated by the clients only. No 'permanent' connection is established by any visualization clients with the state server, which means, the state server does not store or track any information from the visualization clients. The server receive requests from any client connected through the socket and process the specified state operations. The state server handles two kind of operations requested by the visualization clients: set state and get state. Each representational state stored in the state server are int key-value pairs, the key being expressive character string and the value being string serialized from packed Google Protocol Buffers object (discussed below). Each state contains the information of time when the state is created as well as time when the state is lastly updated. The timestamps associated with each state are used to resolve the conflicts resulted from the asynchronism among the different visualization clients.

5.1.2 Data Server

The data server manages all the computation-expensive and data-intensive work and runs on a high performance workstation in a parallel fashion. The server has a two-tiered architecture including a parallel cache and a data processing module. In most

cases, there is only one parallel data server that handles all the related processing work and requests received from the visualization clients. However, for the analysis applications concerning more than one data set, multiple data servers each of which is implemented independently can also dramatically increase the flexibility of the analysis. For example, different feature data sets can be studied simultaneously without the necessity to implement a new data server.

Paraview and VisIt, two well known scientific visualization packages, also work very well in client/server mode. Users can take best advantage of using the high-performance computing ability of the multi-core machine, clusters or supercomputers and interactively conduct the visualization process from office desktops. They could even separate the rendering server and the data processing server such that the users can deploy the rendering work and data processing work to hardware suitable parallel computers. Our data server shares the same strategy.

The data server is implemented using MPI and utilize the master/worker mechanism for dividing up a potentially large computation into smaller pieces of work that is distributed to a pool of workers under the direction of the master controller.

In real application, one may recognize that the server processes except the master controller process which are designed to handle connection from visualization clients and distribute work assignment to the parallel worker processes, very often would stay running with 100% CPU usage. This is caused by the MPI implementation when a MPI process very actively check the communication with other processes.

5.1.3 Visualization Client

Nowadays, many scientific data products are generated to be downloaded, shared and used by domain experts and further analyses are then conducted upon the data sets. Normally, the users can study the data sets from various of angles.

A visualization client handles a specific visual analytic viewport for a certain functionality. It can run remotely on a laptop as well as the same computer with the servers.

There can be as many visualization clients as possible since they are totally independent on each other. They together enable a robust many-aspects data analysis process for domain users. These visualization clients are launched upon users' configuration. As discussed above, the visualization client handles its own special functionality. It coordinates with other visualization clients through the state server and request data directly from the data server. When communicating with the state server, the visualization clients achieve real time signal/event notification using a pull style. The visualization clients actively request the state changes using a specified client-side timeouts.

5.2 Out-of-Core Data Processing

For large-scale data analysis applications, the difficulties and challenges arise from both the scale of data size and multivariate nature of the data variables. Accompanied with the architecture shifts over the years, the primary limiting factor in has shifted from computation to I/O. Childs et al. (2010); Kendall et al. (2011a) I/O systems are typically the slowest component of high performance computing applications. Different from the advanced supercomputing environment, our targeted working environment that is widely accessible to most domain users and researchers gain no benefits from the I/O optimization of the specifically designed parallel file system. Such parallel filesystems include Lustre, GPFS, PVFS and etc that is widely used in supercomputer platforms and also HDFS, one of the popular distributed filesystem in industrial data center. As a result, the I/O performance has become the main bottleneck for such data analysis applications which impedes the interactive process. Instead of deploying the parallel filesystem to the standalone computing workstation or small workgroup computing clusters, we have applied several data management

approaches to improve the I/O performance as well as usability of the ManyView library. In this dissertation work, we have employed a light-weight data partitioning strategy, a well-performed block-based parallel I/O library and a distributed cache to improve the performance.

5.2.1 A Light-weight Data Partitioning Strategy

Block-based partitioning is one of the most popular choices for rectilinear grids. With our targeted analyzing subject being the spatio-temporal observational data sets, block-based partitioning in the 4-dimensional spatio-temporal structured grid is also widely used. However, for a large structured data set, the disk access patterns of analysis application often do not match the physical layout of the data set on the disk, exacerbating the I/O performance. For example, the MODIS data set used in this dissertation work has a high resolution in the spatial dimensions (latitude/x and longitude/y), which is 19968 $\times$ 13568. The data would be stored linearly with the y dimension being the fast-changing dimension. and reading in a small two-dimensional block can result a series of disk search over the whole chunk of space in the data set.

The first strategy to improve the I/O performance is to transform the storage strategy to fit the analysis applications' access pattern. In both of our example analysis applications, random access of relative small spatio-temporal blocks of data set is very common. I have partitioned the data set into relatively small blocks that are stored as separate individual data files ahead of any run-time processing and analysis.

For the MODIS data set, the whole data set (total dimension: 19,968 (X) $\times$ 13568 (Y) $\times$ 506 (T)) is firstly repartitioned into small blocks of data subsets and are stored in NetCDF format. The dimension information of the partitioned block in different granularities are listed in Table 5.1.

For the FNET data set, the whole data set (total dimension: 235,872,000 (T) $\times$ 93 (*Location*), missing data included) is also repartitioned into small blocks of data

Table 5.1: Hierarchy of partitioned MODIS feature data set

Granularity	X Blocksz	Y Blocksz	T Blocksz	Z Blocksz	File Size	Num
0	512	512	64	1	33 MB	8425
1	256	256	64	4	33 MB	2241
2	128	128	64	16	33 MB	561
3	64	64	64	24	13 MB	161
4	32	32	64	32	4.1 MB	49

Table 5.2: Hierarchy of partitioned FNET feature data set

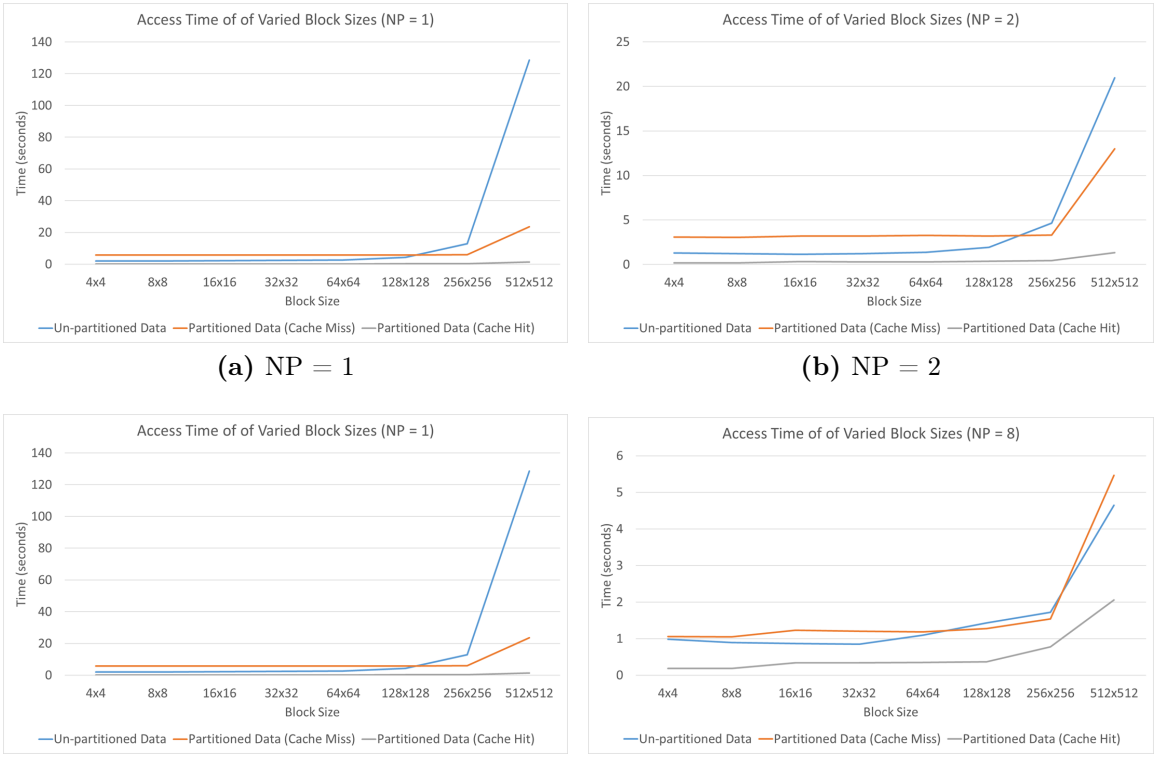
Granularity	T Blocksz	Z Blocksz	File Size	Num
0	864000	1	2.5 MB	15836
1	54000	4	634 KB	15836
2	3375	16	259 KB	15836

subsets and stored in NetCDF format. Due to the characteristics of the FNET data set, the data is a 9-month daily power grid monitoring data set and hence the data set is partitioned in a daily base. The dimension information of the partitioned block in different granularities of FNET data are listed int Table 5.2.

Benchmark

In this subsection, we evaluate the benefit of applying the above proposed data partitioning strategy by the comparison with the performance benchmarks before and after the light-weight strategy. Here we use the MODIS data set for the evaluation. The most general pattern to access the MODIS data set is reading a block of time-varying features. The domain users would select a single or a block of time-varying features for a detailed inspection. In the evaluation, we measured the read access time for a block of time-varying features with their block sizes varied from 4×4 to 512×512 and the number of processes varied from 1 to 8. And the timing result is showed in Figure 5.1. It includes the performance measurement of three types of reading a block of data from the MODIS data set: 1) directly read a block of data from the unpartitioned data sets using BIL; 2) reading a block of data from the partitioned

data sets when cache miss; 3) reading a block of data from the partitioned data sets when cache hit.



(a) NP = 1 **(b)** NP = 2

(c) NP = 4 **(d)** NP = 8

Figure 5.1: Data access time of varied block sizes when executed with 1 to 8 processes.

5.2.2 A Block-based I/O Layer

Block-based partitioning is not only popular in many parallel visualization applications, but also prevalent in other scientific data analysis for large-scale data sets. Kendall et al. (2011a). In addition to the block-based data partitioning strategy. A block-based I/O layer is used to improve the I/O performance. The BIL library developed by Wesley Kendall handles the I/O operations in a collective manner. It greatly reduce the complexity of managing the MPI I/O and low-level system API's using a two-function interface that is easy to use.

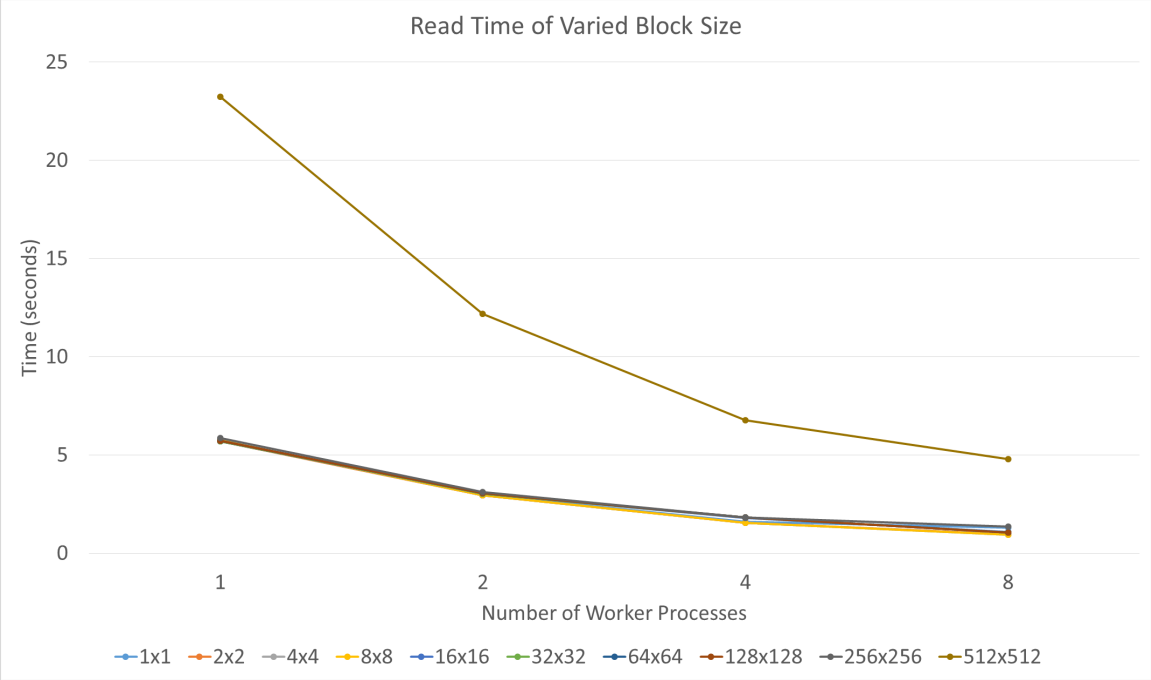


Figure 5.2: Data access time of partitioned data set using varied number of processes.

5.2.3 A Distributed Cache

In order to improve the performance of the analysis process using the system especially to resolve the performance bottle neck resulted from I/O, an extra layer of parallel cache is constructed simultaneously with the data server. The cache module runs in parallel and works in two common modes: 1) caching by work result upon user request. 2) caching by interim data processing output. It largely exploits the parallel processing ability and greatly improve the data processing performance by replace a big part of read and write operation on the hard drive with in-memory operations.

The parallel cache I used in this dissertation work is derived from Dcache, a well-performed distributed cache library developed by Kim Shook. Dcache communicates using MPI (Message Passing Interface). Each MPI rank creates its own cache that is not visible to the other ranks Dcache contains functions for storage and retrieval of raw bytes from the distributed cache. Values may be any number of bytes (less than the total cache size).

The ManyView framework has incorporated the whole Dcache module and customized the data insertion and retrieval pattern to better fit the target analysis application. As the data server is implemented using MPI and it runs mostly with more than one worker process. Different worker processes might request the same copy of data subsets. Hence, the general cache operation API's are listed as following and illustrated in Figure 5.3.

1. dc_get_root(): check the whole distributed cache and retrieve the cache item to the root worker process.

2. dc_get_mask(): check the whole distributed cache and retrieve the cache item to all the worker processes with a non-zero bit mask.

3. dc_insert_root(): insert the item owned by root worker process into the cache and place the cache item to the right worker process.

Benchmark

With the extra layer of the distributed cache, the data access performance has been largely improved in our experiment. We evaluate the performance of applying the distributed cache mechanism discussed in this subsection.

The access of data from the cache are all in-memory operations which are much faster than the same operations from the disk drive. The timing result of both cases is showed and compared in Figure 5.4. For each test of different number of processing cores and block dimension sizes, the performance of accessing data from cache is about 2 to 100 times after than that from disk using BIL depending on the exact execution parameters. Comparing the disk access, retrieving data from the distributed cache does not scale to more processing cores in a single SMP workstation, Tanami in this case. The scaling curve is showed in Figure 5.5. It is because of the fact that the work of collecting, distributing and reordering the cached data among the worker processes incur more overhead. Despite of the relative small overhead, the distributed cache

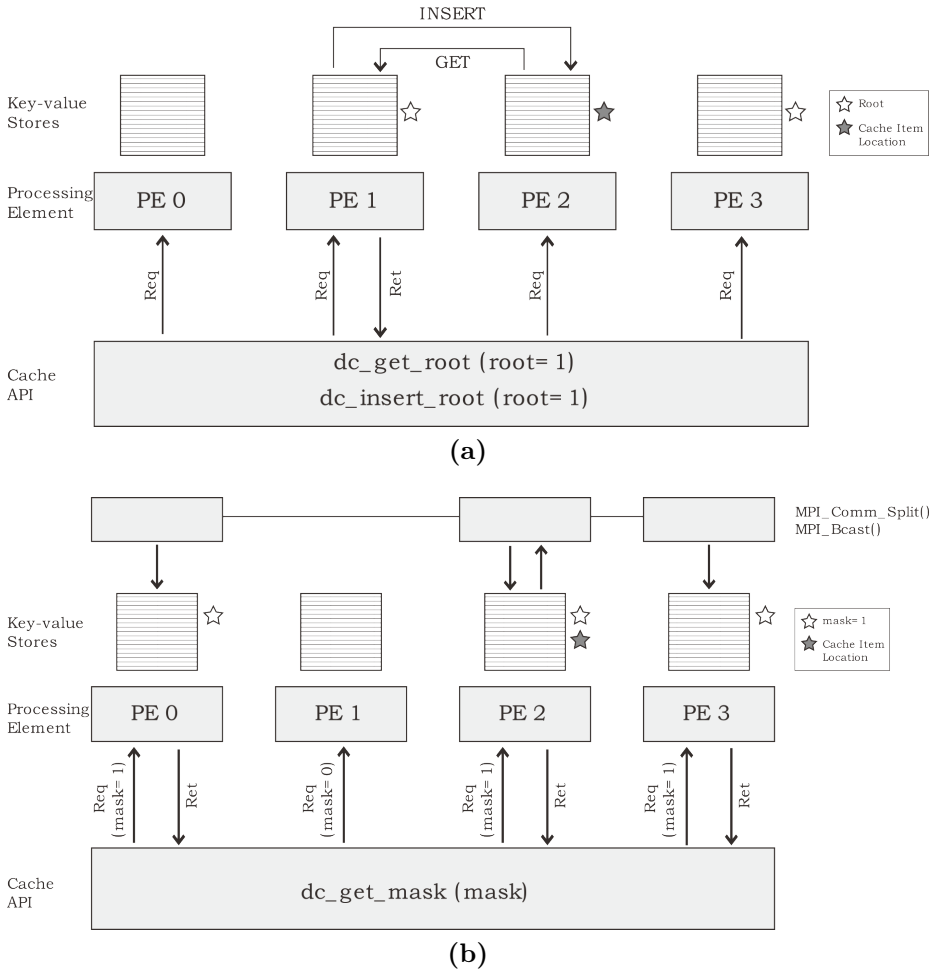


Figure 5.3: The communication pattern of the main API of distributed cache.

performance is stable when executed with varied number of processing cores. We can also expect to see more of the benefit from the distributed cache when the system framework are deployed in a group of computing clusters.

We also evaluate the performance of the distributed cache with respect to the usage information and the result is showed in Table 5.3. The measured timing result is collected in two consecutive rounds of interactive feature selection processes described in Section 6.3.2. The interactions are not exactly the same between two rounds of selection due to the slightly difference of interactive operations including panning and brushing. The data server is executed with a total of 16GB distributed cache and 8 worker processes. We can see an increased cache hit ratio and reduced I/O time as

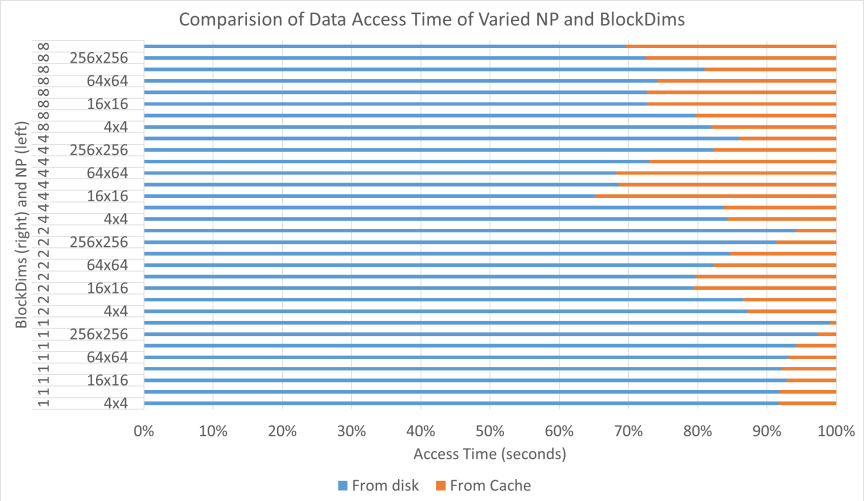


Figure 5.4: Comparison of data access time from in-memory distributed cache vs. disk drive. Each row of the chart represents either the percentage of the data access time from disk (blue) or that from cache (red) within the summed total.

the selection process progresses. Eventually, for a duplicated action, the data server reads all the data from the cache instead of the disk drive (99.40% cache hit ratio).

5.2.4 A Complete Workflow of Data Access

The three data management approaches above make the ManyView framework flexible in accessing data subsets easier and faster by reducing the slow and sometimes

Table 5.3: The usage of distributed cache in a series of example feature selection processes. The whole interactive process includes two consecutive rounds of features selection described in Section 6.3.2. The data server is executed with a total of 16GB distributed cache and 8 worker processes.

Selection Process	Cache Hits (GB)	Total Size (GB)	Hit Ratio	Time (second)
Adirondack Template	3.07	6.89	44.55%	59.69
Pacific Northwest Template	04.29	6.79	63.15%	48.73
Menifee Template	29.15	35.65	81.78%	163.87
Adirondack Template	4.33	4.74	91.16%	17.11
Pacific Northwest Template	6.17	6.30	97.87%	15.10
Menifee Template	13.58	13.67	99.40%	32.51

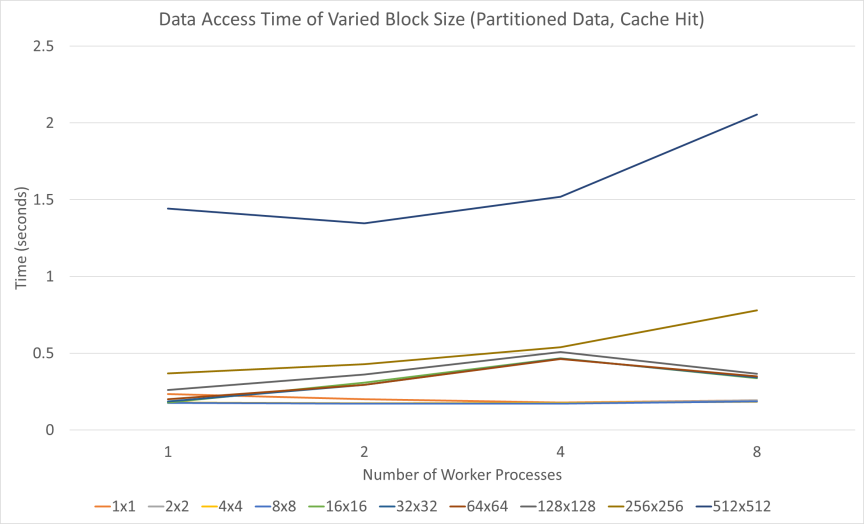


Figure 5.5: Data access time using 1 to 8 worker processes when cache hit.

duplicated I/O work. The whole data set is partitioned into relative small multi-dimensional blocks saved in separate files as discussed in Section 5.2.1. A data structure is used to represent a general data access request for data sets that stored in a structured grid in NetCDF format. The data structure contains multiple block-based information defining the access pattern from disk and cache as well as the layout pattern to the destination buffer. These blocks include the following:

- Target block: the target region of the whole desired data, relative to global data dimensions.
- File block: individual partitioned block of each related data files, relative to global data dimensions.
- Overlapped block: overlapped area between the target block and each file block, relative to the file block.

The whole data access flow is illustrated with an example of a single retrieval request in Figure 5.6. A simpler description of the work flow without the data flow contains a few consecutive steps:

1. Aggregate and distribute data access request;

2. Read data from cache and remove the successful requests;

3. Read data of the remained requests using BIL;

4. Insert the data read from BIL to the cache;

5. Aggregate and reorder the overlapped blocks, then return the data.

With repect to load balancing, all data access requests are aggregated though a gathering and merging process, such that each of the worker processes has a copy of all the requests. These requests are evenly assigned to the worker processes in a round-robin order such that each worker process has roughly the same share of work.

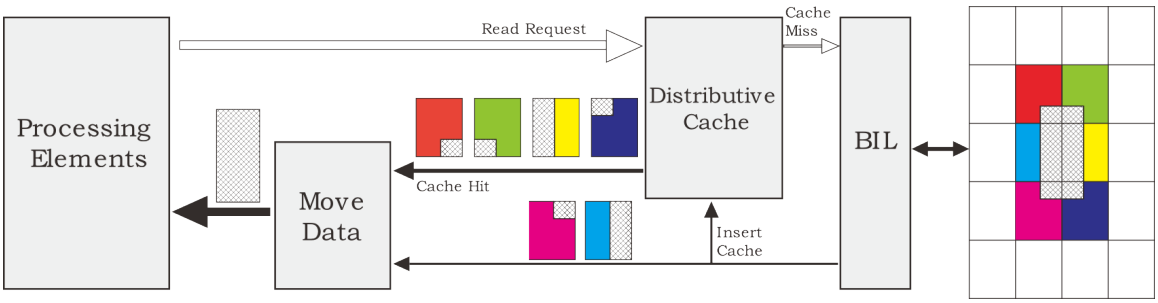


Figure 5.6: An example data flow of data access under the ManyView framework.

5.3 Communication Management

Due to the nature of servers-clients architecture, the communication between the servers and clients is especially important to the ManyView framework. Currently, the connection between the servers and the clients are through the TCP socket. The performance of serialization and deserialization as well as the adaptability and usability for the application users who implement the application-specific functionality at both the data server and the visualization clients sides are the two main affecting factors.

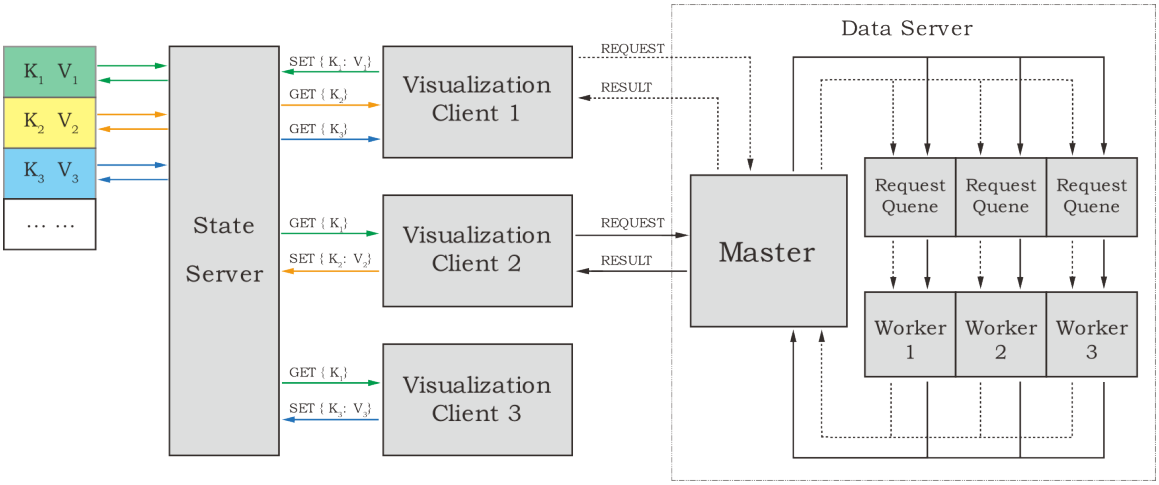


Figure 5.7: Communication pattern between clients and servers

Google Protocol Buffers* meets the above requirement and is used for managing the communication of all the data flow among the distributed components. It is a flexible, efficient, automated mechanism for serializing structured data, users define how they want the data to be structured in a complete protocol buffers object that is easy to be streamed in between the servers and clients. It is also flexible enough to be used together with many programming language with the third-party extensions. As demonstrated in this dissertation work, I have successfully managed the communication in both the C++-based environment and the javascript-based web browsers.

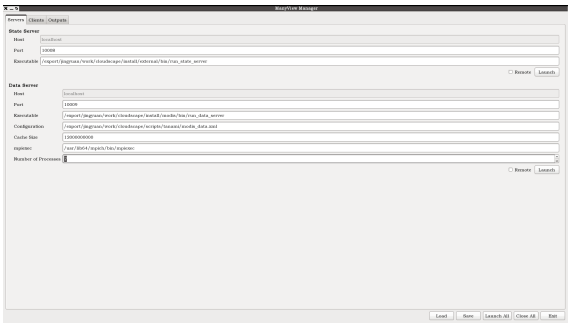
Google Protocol Buffers are not only widely used in the socket communication between servers and clients, it is also largely used to define structured data that is related to the analysis application, and these objects can be very easily packed into a server state or data request/result. Google Protocol Buffers is a very well performed library to create containers for small or simple data objects. The overhead of the generation and the extra memory usage, however, is still not very affordable in very computation-expensive data types, such as the generalized time-varying features.

*http://code.google.com/p/protobuf

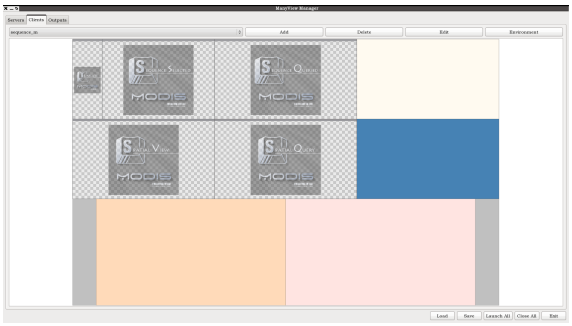
5.4 Task and Resource Management

The ManyView framework is designed to be scalable in terms of both computing resources and number of visual analytics tasks. In order to manage all the resources and tasks effectively and efficiently, a convenient front-end interface is of great help. As part of the ManyView design, XML configuration files are used to encode all the information about the computing resources including the number of processes to execute for each MPI process, the data set configuration including the data path, multi-dimensional resolution and other related attributes or parameters, and the viewport information about the multi-screen setup and displacement for each visualization clients.

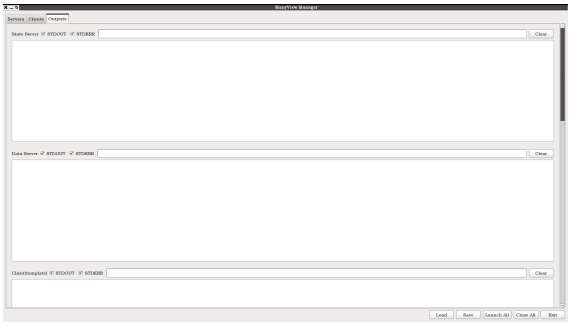
A example task and resource manager as well as a XML configuration is displayed in Figure 5.8.



(a) Server configuration



(b) Client configuration



(c) Output Console

(d) Example XML Configuration

Figure 5.8: Task and resource manager in ManyView framework

Chapter 6

Applications in Situation Awareness

6.1 A General Feature Analysis Pipeline

With respect to a large-scale multivariate observational data set, though the traditional analysis methods and tools are useful, the data processing, visualization, and analysis of them are not directly applicable due to the increasing size, number of dimensions, and complex problem space. More and more challenges that are used to be ignored or underrated in previous studies emerges during the transition to the Big Data era. An efficient pipeline is necessary to enable a desirable way of feature analysis at a deep level in situation awareness applications.

Overall, We propose a general but efficient feature analysis pipeline that includes all the above-mentioned methodologies. Figure 6.1 illustrates the proposed analysis flow. Starting from the large-scale real-valued raw observational data set, several processing and analysis components are divided into two stages: the preprocessing and the runtime interaction stages. At the preprocessing stage, the raw observational data set is abstracted into several multi-scale static feature sets. For different purposes of analysis, the multi-scale static features are evaluated by using certain quantitative feature metrics. They are used for efficient static feature selection or are used to

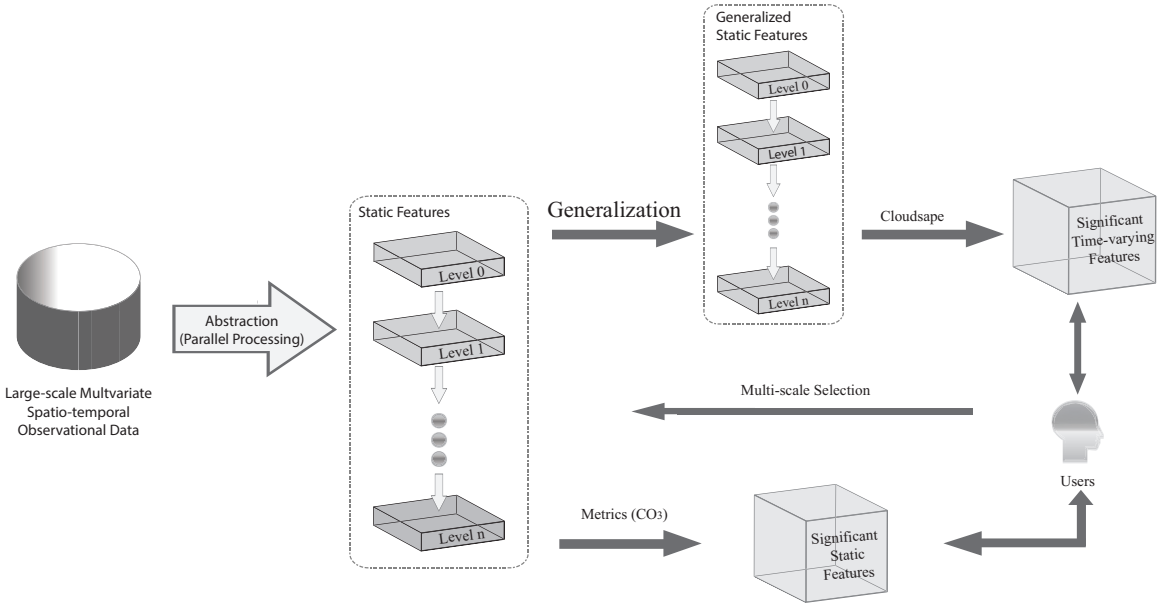


Figure 6.1: A general feature analysis pipeline

generate the generalized static features with the hierarchical granularities well kept. The generalized static feature sets are used for the time-varying features selection.

In this work, the last stage of interactive feature selection is managed by the CO_3 Inspector (Section 6.2.1) and Cloudscape system (Section 6.2.2). Both the CO_3 Inspector and the Cloudscape system greatly reduce users' work by highlighting the important solitary features and, more importantly, group of features. Spatio-temporal similarity distribution are also visualized to assist the domain users in better understanding the magnitude of feature patterns both spatially and temporally. All the informative cues, for example, visual hot spots along with the interactive feature selection process, provide further guidance to users to carry out iterative analysis that leads to potential new scientific discoveries. The usefulness of the feature selection systems is demonstrated through the examples in both driving applications in Section 6.3 and 6.4 respectively.

6.2 Coordinated and Distributed System

We have implemented two complete interactive systems to engage users with a efficient interaction and iterative feedback in the analysis process of selecting both significant static features and dynamic time-varying features. CO_3 Inspector combines adequate statistics information with CO_3 metrics in the analysis framework. Cloudscape provides flexible methods to generate meaningful and purposeful feature selection templates, and it is used to select time-varying features to interest at run time. CO_3 Inspector and Cloudscape together comprise a series of incremental research endeavors. The Inspector is highly coordinated among different analysis viewports, and Cloudscape has taken a further step and integrated ManyView system framework that fully exploits the computing and displaying resources of the targeted environment, which makes it a highly coordinated and distributed system. In this section, I will articulate the implementation of the two user interfaces.

6.2.1 CO_3 Inspector

In our static feature selection application using CO_3 metrics, the raw data is abstracted into static features through a parallel hierarchical clustering algorithm. All the static features, or clusters, which are generated by hierarchical clustering algorithm, are evaluated and preprocessed into imageries and used for user interactions to select significant features.

Figure 6.2 shows the initial view of CO_3 Inspector for MODIS. The interface has three components: a spatio-temporal view, a co-occurrence graph, and statistics plots.

All the static features are assigned to different colors based on cluster centroids, and the same color scheme is used across all views and clustering levels. Since the system is designed for visualizing a large number of static features and the color represents the multivariate properties of the static features rather than categories, repetitive color assignment, as used in Dimstiller Ingram et al. (2010), is not an option in this case. In MODIS, the color-map is indexed according to the primary

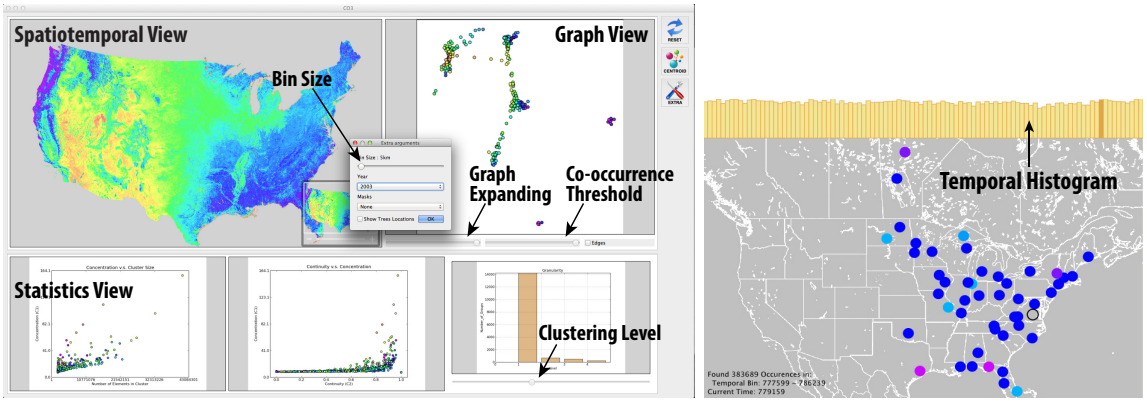


Figure 6.2: (Top) A snapshot of CO_3 Inspector showing the spatio-temporal view, graph view, and statistics view; (bottom) Spatio-temporal view adapted for the power grid application.

and secondary principle components. In FNET, the red, green, and blue channels are assigned according to the changes in frequency, voltage, and phase angle, respectively. Missing data is transparent here.

Spatio-temporal Rendering

The spatio-temporal view is specifically designed for different data sets. For MODIS, 2D image-based rendering is implemented, while FNET uses an adapted view with a temporal histogram above the map and sensor locations represented by the colored disks. In the temporal histogram, the entire day's data is partitioned into roughly 15-minute intervals, and the height of a bar corresponds to the number of occurrences of chosen static features during the 15-minute interval.

In both cases, spatio-temporal renders color at each location according to its cluster membership.

Co-occurence Graph

We use a graph layout to visualize co-occurrence. At any level of the cluster hierarchy, we can consider each static feature as a node v in a graph $G(V, E)$ with edge weights assigned by the C_3 metric.

The graph layout is computed using a force-directed method with an energy barrier Martin et al. (2011). Proximal static features are represented as proximal nodes in the final layout. Also, we capture the animated process during which a graph layout converges. Users find that the functionality of being able to view at least the final steps of a converging graph layout becomes very useful in examining subtle differences in co-occurrence. This is demonstrated in Section 6.3.1.

Statistics Plots

In the data exploration process, statistics is a classical way for domain experts to explore local or global characteristics of data. When coupled with complex rendering techniques, this numerical exploration can effectively assist in the user interaction. In our application, the statistical view offers an easy and flexible interface to control the multi-levels of clustering results, and its quantitative feedback is often helpful. CO_3 Inspector provides four widgets: a scatterplot widget of C_1 *vs.* C_2, a scatterplot widget of C_1 *vs.* feature size, a histogram of the number of static features in any hierarchical level and a parallel coordinates plot activated upon selection of static features in any space (Figure 6.17(e)). The parallel coordinates plot is used to display the multivariate values of cluster centroids.

Multiple-view Coordination

Every view in the interface is fully coordinated with all other views such that any actions taken in one view is immediately reflected on all others. In this context, analysis is an iterative and user-driven approach through that each step provides instant feedback while refining focus.

During the interactive visualization phase of CO_3, only static features are analyzed. It skips over the raw multivariate time-varying data. Brushing is enabled to select static features in any of the viewports. Selected features are highlighted with a semi-transparent plus sign. Brushing using the left mouse button makes 'fresh'

selections whereas brushing done with the right mouse button selects a subset of the already selected features. Brushing operations can be arbitrarily chained together as a result of iterative user interactions.

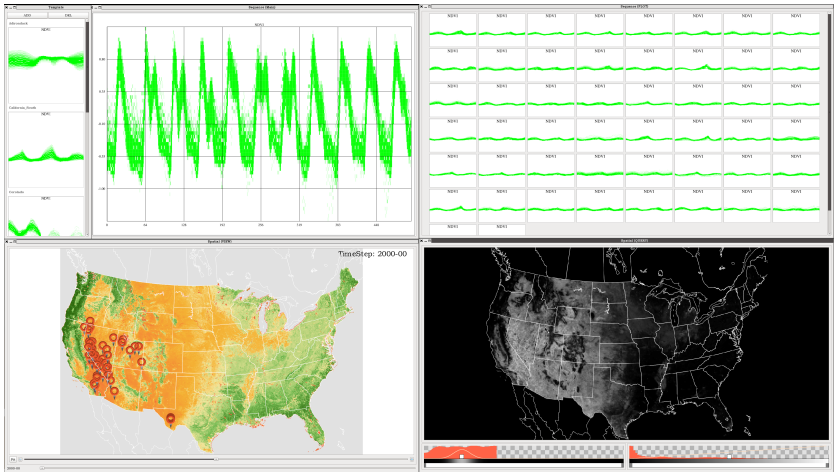
Implementation

The Inspector System employs image-based rendering techniques. Matplotlib, a python plotting library, is used to generate statistics plots and co-occurrence graphs. They are pre-generated only once after the feature extraction and evaluation of the CO_3 metrics. Such preprocessing improves the speed of interaction and provides the system with comprehensive plotting features. The Inspector System then offers visualization functionalities interactively to provide immediate feedback on a single laptop computer. The rendering preprocessing, including the calculation of the co-occurrence graph layout, is executed in parallel. This takes about 7 minutes for FNET and 60 minutes for MODIS on a 12-core Linux workstation in the setting presented in the paper.

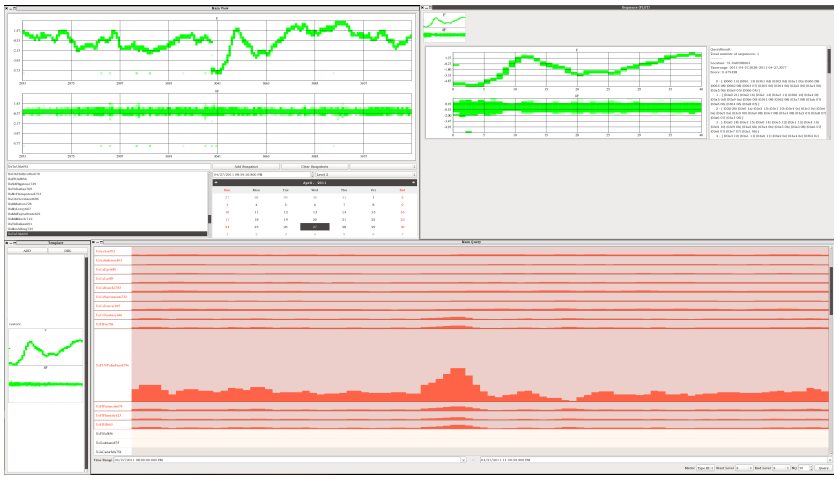
6.2.2 Cloudscape

The Cloudscape System aims at providing a flexible way to effectively select time-varying features based on the needs and interests of users. The system has entirely incorporated the ManyView framework and therefore is comprised of multiple independent visualization clients. Figure 6.3a and Figure 6.3b are snapshots of all the visualization clients for both MODIS and FNET data sets.

The Cloudscape System mainly contains the visualization front ends described in the following subsections. Each of the visualization clients is responsible for specific visual analytic tasks, both interaction and presentation. These clients coordinate with each other and retrieve raw and runtime processed results through the ManyView framework.



(a) Cloudscape for MODIS



(b) Cloudscape for FNET

Figure 6.3: Cloudscape screen shots for both MODIS (top) and FNET (bottom) data sets.

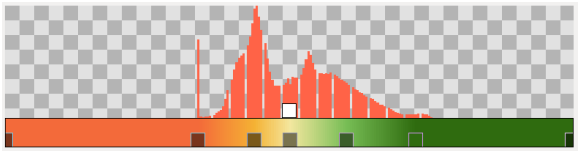
Template Manager (MODIS & FNET)

As we discuss in previous section, users define their interest of time-varying features by creating a time-varying feature template. The Template Manager handles the creation, organization and visualization of time-varying feature templates to be used as the target of feature selection. It communicates with the state server to retrieve the coordination states that is necessary for creating a template such as user interaction of highlighting example feature sets, modify the active feature template for on-going

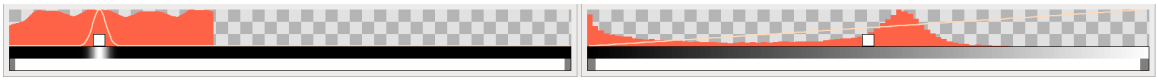
feature selection tasks, and etc. Users are able to create the template by online generation from example feature sets, from direct sketching in variable space, and from the pre-generated template stored as files on disk drive.

Spatial Rendering (MODIS)

Spatial Rendering, a visualization client that handles the direct rendering of the static features at a specific time step, plays a very similar role with the Spatio-temporal Rendering View (Figure 6.2) in CO_3 Inspector system, but it only applies to the MODIS data set. As the static features are stored in a multi-resolution hierarchy, the visualization client dynamically determine the best data granularity level and the visible spatial region at run time and request a minimal sized but visually precise subset of the data from the data server. With the multi-resolution rendering strategy and the data management approaches of the ManyView framework, especially the distributed caching, the users are able to visualize the animated temporal variation at the client side.



(a) Spatial Rendering's transfer function widget



(b) Similarity Distribution's query widgets

Figure 6.4: Color and opacity widgets in different visualization clients

The colors of the static features are determined by the central NDVI values of features (centroid of the quantized variable range of each static state) and the rendering transfer function that is flexibly controlled by the interactive widget (Figure 6.4a). The color scheme applied in Spatial Rendering visualization client in the rest of the work is mainly determined based on a five-color combination

Figure 6.5: Color choice for rendering

(Figure 6.5) that is picked up from Adobe Kuler*, which visually describes the transition from desert to forest well.

Highlighted Features (MODIS)

In Cloudscape for Modis, whenever the users interactively highlight a single time-varying feature by picking or multiple time-varying features by brushing in the Spatial Rendering visualization client, the Highlighted Feature visualization client updates its feature view with the rendering of the up-to-date feature(s).

Temporal Rendering (FNET)

The Temporal Rendering visualization client manages the interactive visualization of the time-varying features for FNET data set. Using a calendar-based widget and a list of locations where the data are measured and collected, users are able to highlight and visualize with the time-varying features in the feature viewer.

Selected Features (MODIS & FNET)

The Selected Features visualization client displays all the selected time-varying features in a standalone viewport. All the features are displayed in tiled thumbnails, and each of which is clickable for a focused visualization in a embedded feature viewer.

*https://kuler.adobe.com

Similarity Distribution (MODIS & FNET)

A typical time-varying feature selection process using Cloudscape system returns not only the selected features which are displayed in Selected Features visualization client but also the spatio-temporal similarity distribution.

In Cloudscape for Modis, the Similarity Distribution visualization client contains three components: spatial similarity mapped view, temporal similarity view (Figure 6.4b) and similarity histogram (Figure 6.4b). Users can utilize the temporal similarity view and similarity histogram to control the subset of the spatial similarity distribution.

In Cloudscape for FNET, the Similarity Distribution visualization client renders the temporal similarity distribution of the time-varying feature selection process. Users can flexibly choose the subset of feature sets in which the time-varying feature selection is conducted.

Implementation

The Cloudscape System fully incorporated the ManyView framework and A parallel data server implemented by using MPI and pthread as well as a state server implemented through using pthread are executed as the processing and coordination backends. All the visual analytics tasks that the time-varying feature templates are created, the selected features and spatio-temporal similarity distribution are visualized, and etc., are implemented as separate visualization clients. Most of the clients is implemented by using Qt, and the connection between the visualization clients and the servers are handled in a separate thread in order to have a smooth interaction.

Different from the CO_3 Inspector, the Cloudscape System does not rely too much on pre-generated data and imageries except the hierarchical feature sets from initial feature extraction procedure. All the processing works are executed on the fly at a near interactive rate.

6.3 Characterizing Phenology of Forest Ecosystems

Users can benefit from the interactive systems in their analysis of features. Using CO_3, users are able to select significant features based on the quantitative CO_3 metrics.

As discussed in Section 3.1, the NDVI values are preprocessed into data abstractions, widely named as static features in this work. The domain experts have defined **PHENOLOGY** as a study of the periodic events in the life cycles of animals or plants, as influenced by the environment (especially seasonal variations in temperature and precipitation). The features extracted from the MODIS data set that are used to select significant static features using CO_3 are the hierarchical clusters, each of which represents a phenology class that provides information of specific vegetation growing pattern year-wide and is termed as **PHENOSTATE** by Kumar et al. (2011a).

6.3.1 Selecting Significant Features Using CO_3 Metrics

Using the statistics view widgets of the CO_3 Inspector, users are able to specify significant features by selecting the strong or weak properties of the CO_3 metrics.

Example 1:

Figure 6.6 shows an example of one unique feature extracted from the MODIS data set. View A in Figure 6.6 shows the concentration *vs.* continuity space while View C in the same figure shows the concentration *vs.* feature size space. In both views, dozens of phenostates stand out from the whole population in the space and spotting them is straightforward. The selected one (in red) is small but has a relatively high concentration and continuity. The map in View D shows the geographic locations with the phenological properties defined by the phenostate selected. The three labeled regions in the figure represent salt flats and areas with white sands. The Bonneville Salt Flats near the Great Salt Lake is the most contiguous and concentrated feature. The other areas including the White Sands National Monument capture similar phenological properties— areas of the United States that remain a

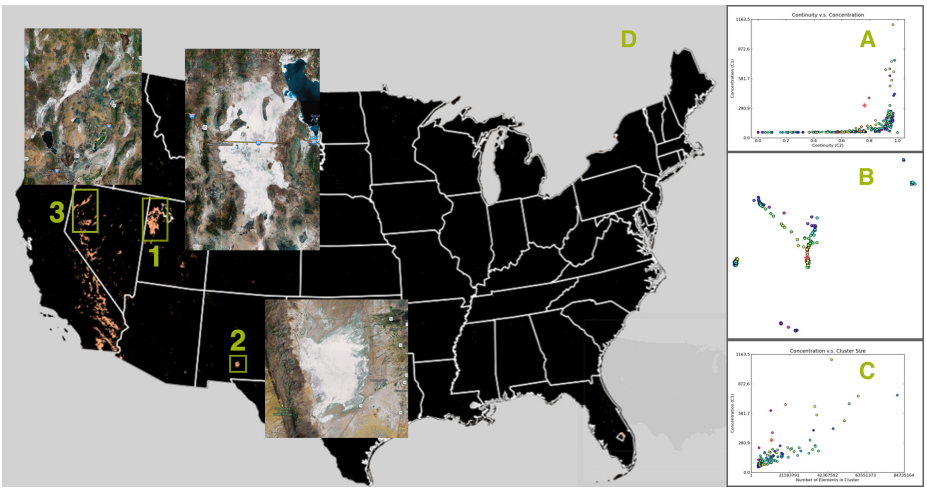


Figure 6.6: A highly continuous and concentrated feature in the MODIS data that captures areas of salt plains and white sands - The Bonneville Salt Flats (1), White Sands National Monument (2), and other salt flats (3) are highlighted. (Year 2000, 15km bin size)

very white color year round (in contrast with snow, which is seasonal) and have absolutely no vegetation.

Example 2:

Exploring the co-occurrence graph relating to the salt flat phenostate mentioned above, we find a related feature group spreading out sparsely over the whole continental U.S. By further drilling down on some of the co-occurring phenostates, a set of small phenostates shows an interesting spatial distribution illustrated in Figure 6.7. The three labeled areas in Figure 6.7 are representative of arid lands that contain significant green areas because of human activities and irrigations. These static features are phenologically similar to the salt plains in that they are regions with a severe lack of available water.

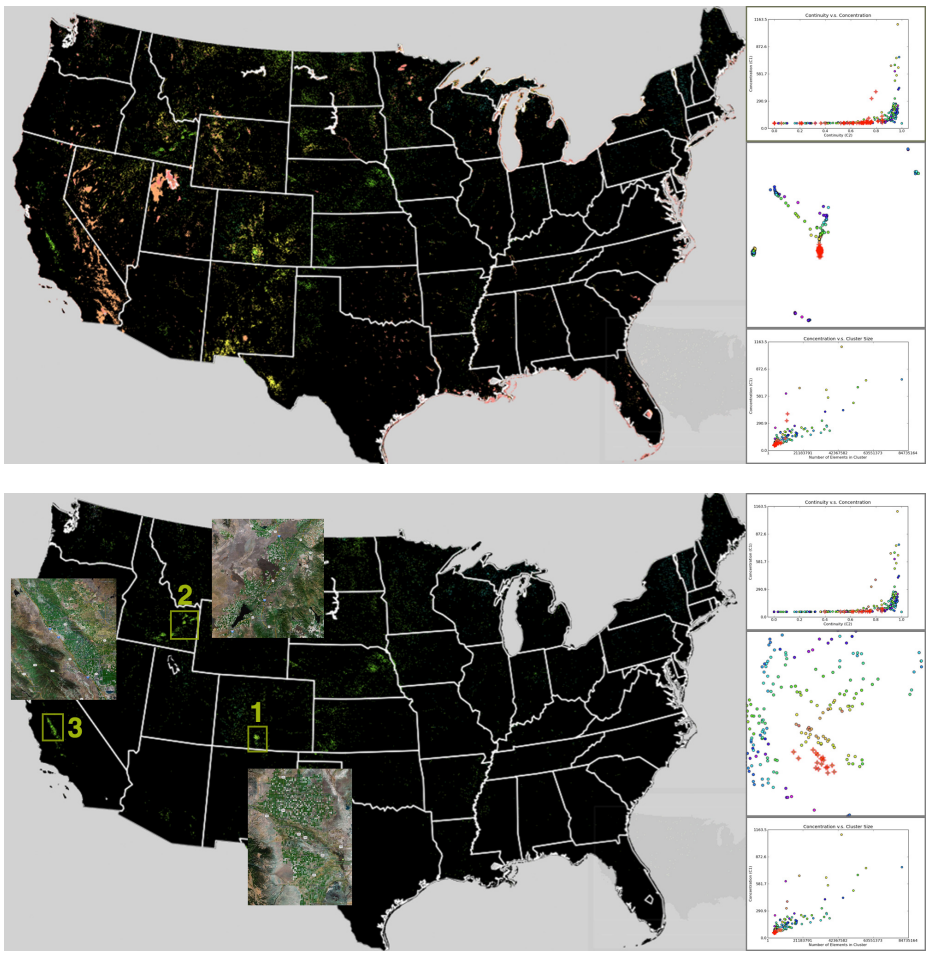
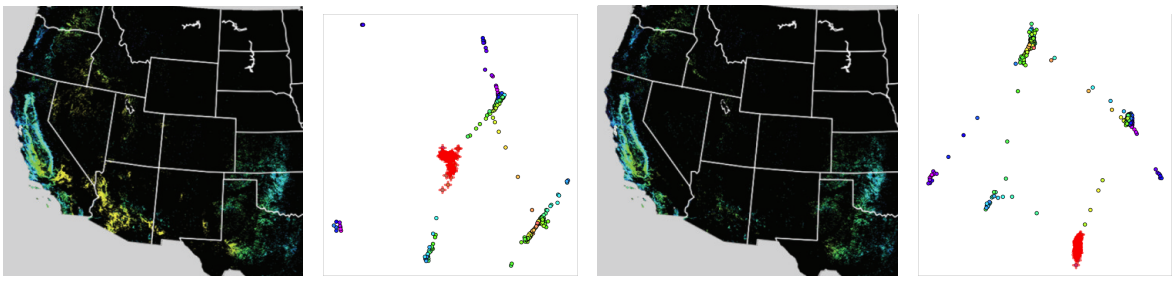


Figure 6.7: An example feature group in the MODIS data. The selection starts from the example feature of salt flats and white sands. After a series of navigation and selection refinement steps, users are able to discover such a feature group. These phenostates reside in some large irrigated lands in dry areas. (Year 2000, 15km bin size)

Example 3:

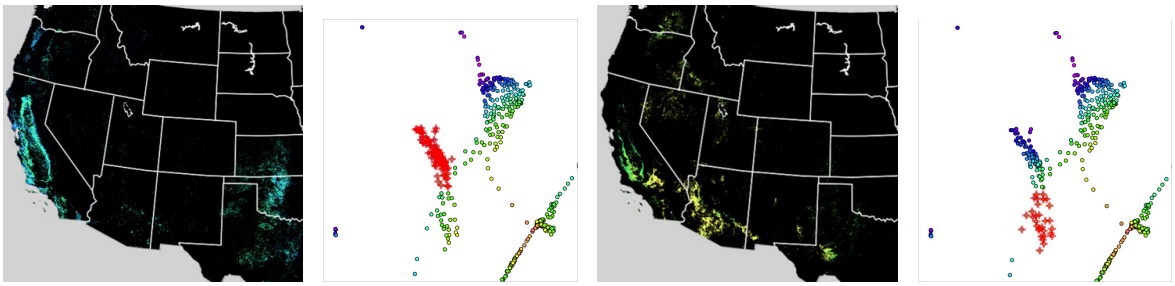
Starting from the co-occurrence metric in the MODIS data set with a 5km bin size, we notice a feature group in the co-occurrence graph which is highlighted in the spatial view (Figures 6.8(a) and 6.8(b)). It appears that this feature group represents the outline of the Central California Valley and other areas in the Southern Great Plains, both of which reveal rather irregular growing patterns due to the terrain of the Sierra Nevada mountain range and the highly varied weather patterns in the southern part of the Great Plains. By observing an earlier stage of the graph layout convergence process, as shown in Figures 6.9(b) and 6.9(d), we can see that the original feature group is now expanded into two distinct areas. The top half of the group gives a near-exact outline of the Central California Valley.

We iteratively adjust the bin size to 20km (Figures 6.8(c) and 6.8(d)). We notice that the group of static features gets tight. However, the two main parts in the original group of features are separated when the bin size is 20km. It renders the Central California Valley directly selectable without going back to the earlier layout process. According to this result, with larger bin sizes, we are able to select structures belonging to larger spatial scales.



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

Figure 6.8: An example feature group in the MODIS data. The spatial distribution a the feature group containing areas in the Southern Great Plains and the outline of the Central California Valley (Year 2003, (a, b) 5km bin size, (c, d) 20km bin size).



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

Figure 6.9: We expand the graph in Figure 6.8(b) and refine the selection to two individual parts that initially appeared together. The area showed on the top row is almost totally correlated to the Central California Valley. (Year 2003, 5km bin size)

6.3.2 Selecting Significant Time-varying Features Using Cloudscape

With the help of the Cloudscape System interface, domain users can conduct feature analysis in an efficient and iterative way. In this section, four examples of time-varying feature selection are illustrated either to demonstrate the general interaction of the system of Cloudscape for MODIS or to present the typical types of analysis procedures of selecting significant time-varying features , specifically temporal phenology patterns in the MODIS data set.

The yearly greenness observations have been the central research target by domain scientists. Kumar et al. (2011a)'s parallel k-means clustering as well as our similar practice in Section 3.1 of applying hierarchical clustering to abstract the raw NDVI data set into phenological static features are both based on the annual NDVI values. As a result, most of the following examples are based on the time-varying feature template exactly lasting for one year. However, using the Cloudscape System, the selection of interesting or significant time-varying features does not restrict to the single-year features that start at a fixed time, for example, January 1 of each year, which is determined and not changeable after the data abstraction process. Users are able to flexibly create time-varying feature patterns more than or less than a year and the selected result are also not restricted to start and end at the same period of

time. As long as the feature patten matches well with the template pattern, it can reside at any spatial location, start, and end at any arbitrary time steps.

Example 1

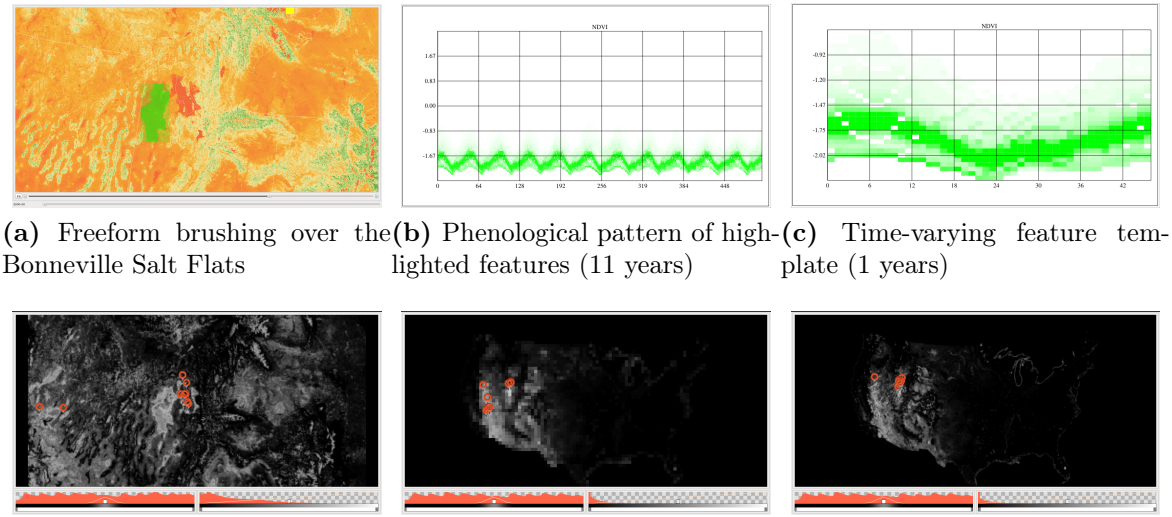
Similar to the first example presented in Section 6.3.1, we take the unique phenological pattern exhibited in the salt flat in Utah state for example and the starting point and also target of a time-varying feature analysis.

In one of the most common practices of analysis, users started from accumulated knowledge about time-varying features. As discussed and demonstrated in Section 6.3.1, the Bonneville Salt Flats near the Great Salt Lake has a very unique phenological pattern - white year round. In Figure 6.10b, the 11-year time-varying feature pattern is directly composited by all the highlighted features. The vegetation indices stay mostly between -2.5σ and -1.5σ, but still present a seasonal change each year similarly. Figure 6.10c sketches the generated template time-varying feature pattern.

User can use the brushing tool implemented in Cloudscape system and highlight the exact shape of the region as the base for generating feature selection template (Figure 6.10a). The freeform brushing defines the group of example time-varying features being extracted directly from the data set. These time-varying features roughly present a similar variation pattern and the generated feature template becomes an abstracted representation of the whole salt flats.

The spatial similarity at two different data granularities and two different feature selection strategies are compared and illustrated in Figure 6.10d, 6.10e and 6.10f. The rendering of the spatial similarity distribution applys a linear gray-valued color map where the darker of the fragment color the less similar with the feature template. Figure 6.10d is the matching similarity distribution in the neighboring regions of the Bonneville Salt Flats, where the selection template is created at the same granularity. And the selected time-varying features most matched with the defined interest by the users also include several smaller regions located in West Nevada. The distribution

rendering depicted the regional difference well based on the degree of similarity with the phenology pattern created. For such a time-varying feature which describes an phenological pattern that has little vegetation, the selection result of the spatial similarity distribution are showed in Figure 6.10e and 6.10f. One applys the coarse-to-fine strategy and the other select the time-varying features at a fixed granularity which returns a finer similarity distribution. The coarse-to-fine feature selection strategy is mostly suitable for a feature selection at a large geographical region of which the amount of data and processing work at the finest granularity is not affordable for a runtime interaction. The multi-scale alignment of target time-varying features and the template feature speedups the selection process and still provides a informative distribution result as showed in Figure 6.10e.



(a) Freeform brushing over the Bonneville Salt Flats

(b) Phenological pattern of highlighted features (11 years)

(c) Time-varying feature template (1 years)

(d) Similarity distribution near the Bonneville Salt Flats

(e) Similarity distribution of CONUS (coarse to fine)

(f) Simialrity distribution of CONUS (fine)

Figure 6.10: (a)A time-varying feature template is created based on a group of time-varying features at the Bonneville Salt Flats. The template describes a highly unique phenological pattern of being yearly white. At both granularities and using different feature selection strategies, users are able to visualize very informative regional distribution of the designated feature pattern with varied level of details.

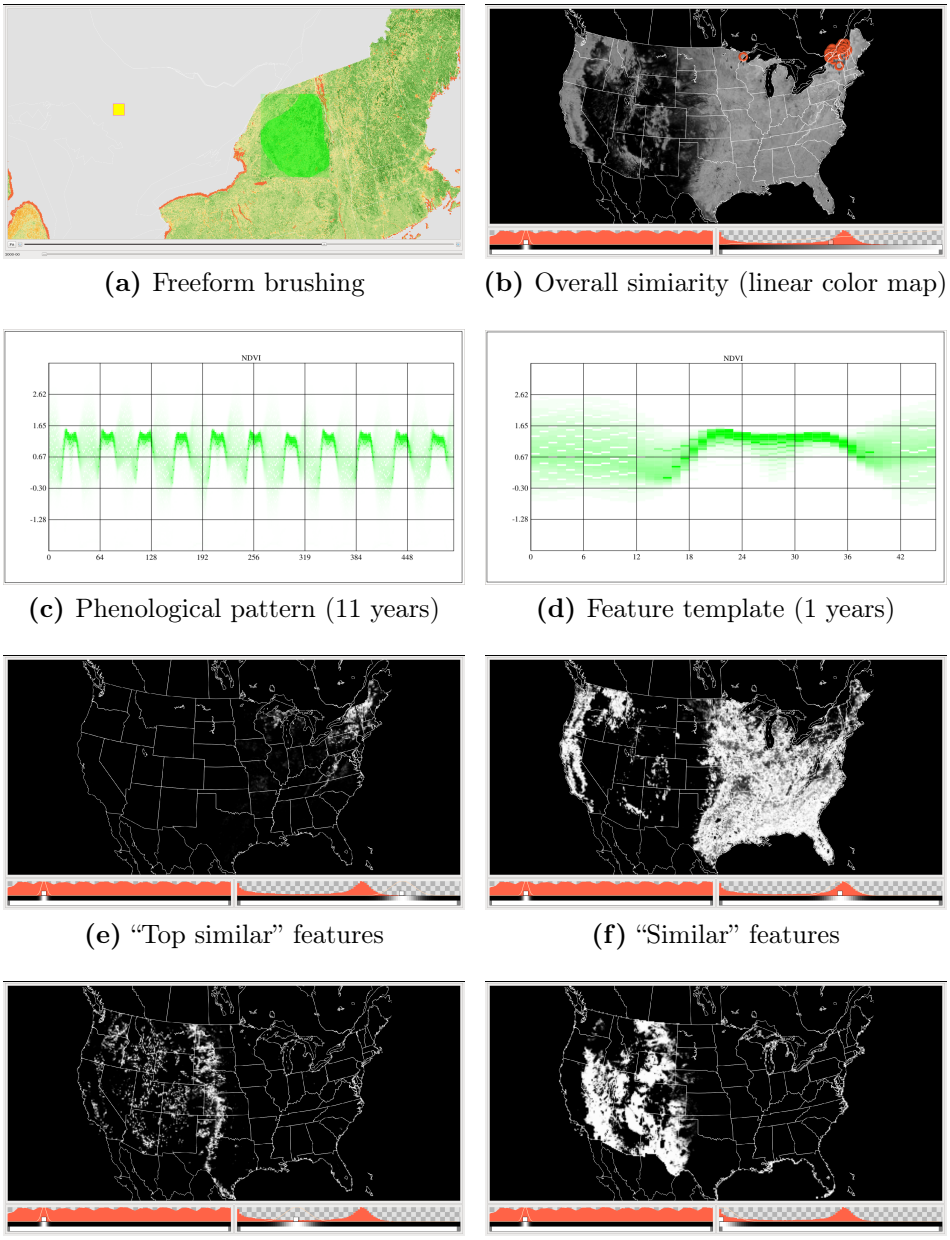
Example 2

As illustrated in Example 1, the process of feature selection does not only find the most significant time-varying features similar to the user-created feature template but also reveals the similarity distribution of the template in the spatio-temporal space. It is especially useful for domain users to study the regional pheonology comparison at large scale. Additional to the overall view of the similarity distribution, users are able to visualize different subsets of whole population of features in selection, based on the degree of similarity. For example, among all the target time-varying features, they can be divided into groups of "top similar", "similar", "more or less similar" and "different". The division is fuzzy using the Gaussian controller in the similarity score histogram in the interface and is totally managed by the user interaction.

In this example, we start by interactively brushing a freeform selection in geospatial domain and the time-varying features in the highlighted location and time range are used as the input to create the feature template. The feature template is used to select significant features over the whole continental United States and the similarity distribution of four subsets of all the target features are rendered and compared for better understanding of the phenology of both the highlighted location as well as the whole United States.

We pick three representative locations, Adirondack Mountains, Great Smoky Mountains and Pacific Northwest Coast, and conduct the same above-mentioned selection process. Each of Figures 6.11, 6.12 and 6.13 illustrates the interactive brushing of creating the feature templates and their corresponding similarity distribution. As showed in the Figures 6.11c, 6.12c and 6.13c, the yearly phenological patterns at each of the locations stays stable in the 11-year time span. Therefore, we have taken the year of 2000 as the input for template creations. Comparing the three phenological pattern described by the feature templates of the three locations, we are able to recognize their uniquenesses. All of the locations are considered to be seasonal green as we can tell their NDVI values are around or higher than the CONUS average (0σ).

Adirondack Moutains and Great Smoky Mountains presents quite similar pattern except Adirondack Mountains has a shorter period of time between two overall peaks of "greenness". The phenology pattern of the Pacific Northest Coast presents a much smoother variation pattern and unlike the other two locations, the peak of "greenness" comes after early March instead of summer time. In Figure 6.11e, 6.12e and 6.13e, we restrict the rendered spatial similarity distribution on only those most similar ones. Alongside with the locations where the feature template is created from, we can also identify a few other locations that has a similar phenology pattern. For example, the Pacific Northwest Coast feature template help select a few good matches far away along the Southeast Coast. The comparison of the similarity distribution of subsets of features assist in the understanding of the corresponding phenology pattern and the regional differences. It is even more intuitive to visualize the animation of the distribution rendering by changing the Similarity Gaussian Controller's central matching score.



(a) Freeform brushing

(b) Overall simiarity (linear color map)

(c) Phenological pattern (11 years)

(d) Feature template (1 years)

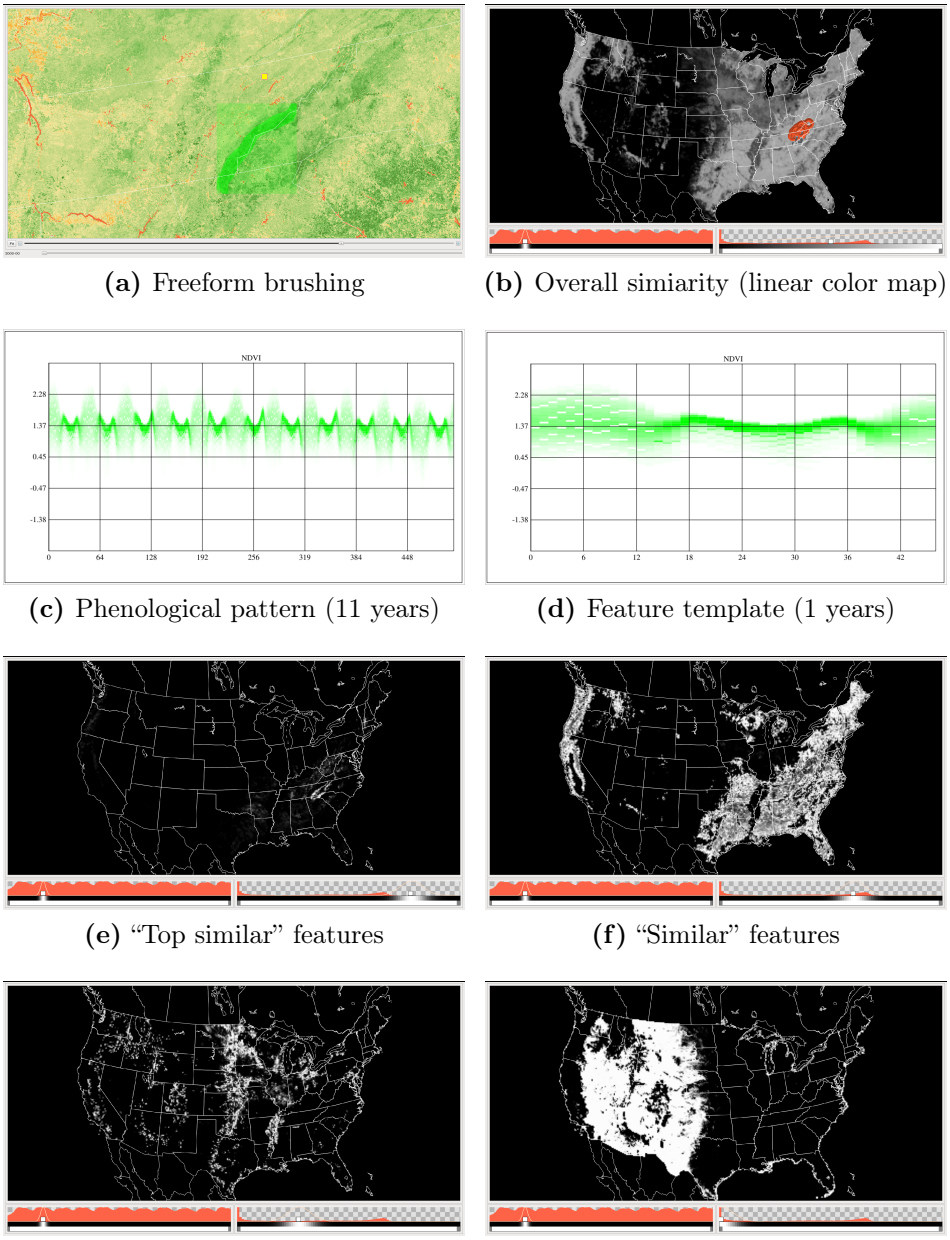
(e) "Top similar" features

(f) "Similar" features

(g) "More or less similar" features

(h) "Different" features

Figure 6.11: The feature selection result using a time-varying feature template created based on a group of time-varying features at the Adirondack Mountains. (e, f, g, h) shows the comparison of spatial similarity distribution of four subsets of target features.



(a) Freeform brushing

(b) Overall simiarity (linear color map)

(c) Phenological pattern (11 years)

(d) Feature template (1 years)

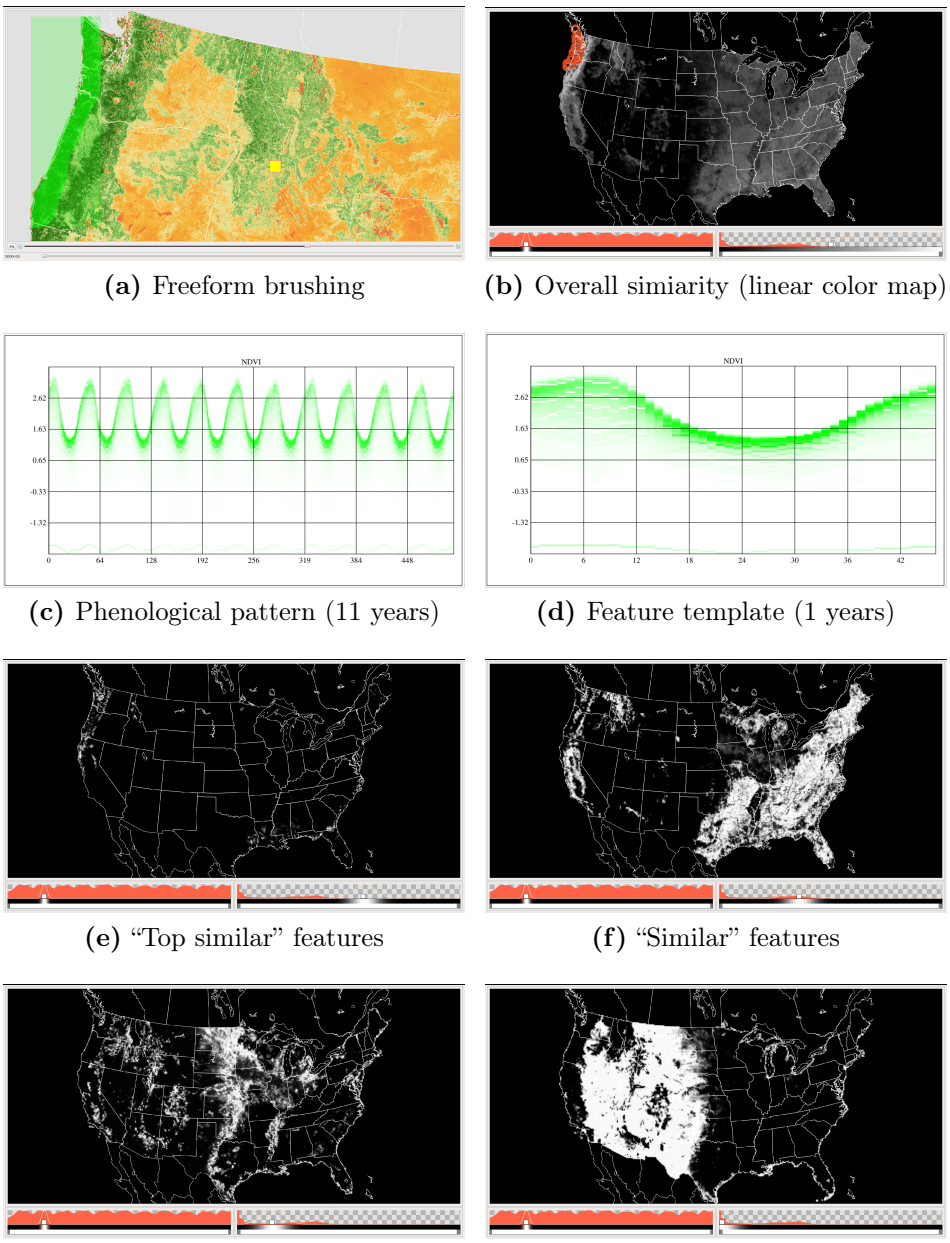
(e) "Top similar" features

(f) "Similar" features

(g) "More or less similar" features

(h) Different" features

Figure 6.12: The feature selection result using a time-varying feature template created based on a group of time-varying features at the Great Smoky Mountains. (e, f, g, h) shows the comparison of spatial similarity distribution of four subsets of target features.



(a) Freeform brushing

(b) Overall simiarity (linear color map)

(c) Phenological pattern (11 years)

(d) Feature template (1 years)

(e) "Top similar" features

(f) "Similar" features

(g) "More or less similar" features

(h) Different" features

Figure 6.13: The feature selection result using a time-varying feature template created based on a group of time-varying features at the Pacific Northwest Coast. (e, f, g, h) shows the comparison of spatial similarity distribution of four subsets of target features.

Example 3

As demonstrated in the first two examples, the Cloudscape system enables the domain users to examine their possessed knowledge about certain feature patterns and study its spatio-temporal distribution in similarity. Additionally the system also provides a very flexible analysis environment. Users are able to select significant or interesting time-varying features not only by the direct selection result but also in the iterative process in which the users are guided to an unfamiliar spatial region or unnoticed phenological variation pattern. An example of such process is illustrated in Figure 6.13 and 6.14. In the Example 2 above, we have generated a time-varying feature template based on pattern within the Pacific Northwest Coast area. The area exhibits a quite unique phenological characteristics as we can tell from the limited amount of high scores in the similarity histogram. Most of its similar time-varying features are still near the Pacific Northwest Coast while we can also identify a few locations along the southeast coast where the "top similar" features exists. These locations are obvious enough to draw users' attention and further inspection of these locations at the more detailed granularity becomes a nice and easy-to-make choice. Out of these locations that spread over multiple near-coast states, this example has focused one of the locations in North Carolina State after a drill-down interaction and visualization. The location (marked in Figure 6.13e) is not very easily noticeable in a regular interaction process due to its relative small size.

The static rendering of the region are showed in Figure 6.14. One can quite easily differentiate several small regional segments of the area on the static features. Using the interactive brushing tool, we can examine the detailed of the time-varying feature patterns exhibited at any of these segments. Three representative time-varying feature pattern is interactively highlighted and the result composited renderings of the highlighted features are showed in Figure 6.14d, 6.14e and 6.14f. Comparison among these features provides direct knowledge about the local phenology. Out of the three highlighted regions, both Figure 6.14e and 6.14f are very similar to the

Pacific Northwest Coast feature template in the first 8 years (2000 2007). Beginning from the 9th year, the small region highlighted in Figure 6.14b has a significant drop of the "greenness" measurement and recovery continuously later on. We did not conduct further analysis here, however, the abnormal change in the annual phenology could well be used for more in-depth study by domain experts or interested users.

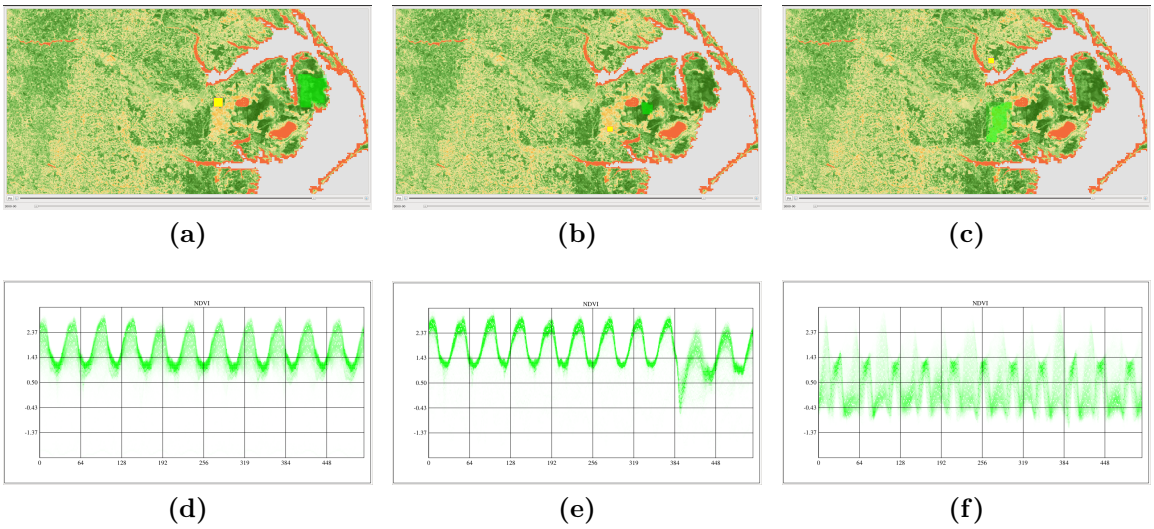


Figure 6.14: Phenological pattern of three small highlighted regions. Parts of the location contains "top similar" phenological pattern selected by the feature template created in Example 2 and illustrated in Figure 6.13. (Top) freeform brushing of the representative small regions. (Bottom) Phenological patterns (11 years).

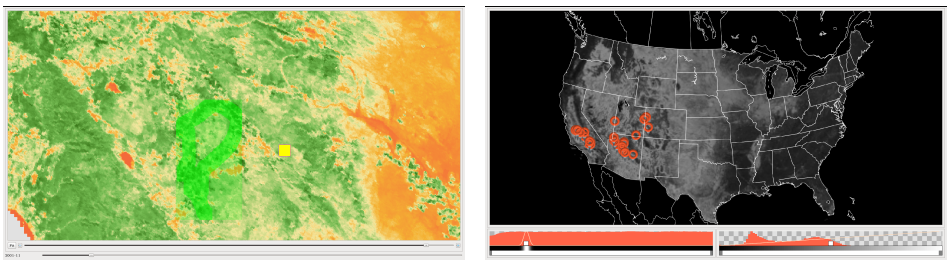
Example 4

This example illustrates the feature selection using a multiple-year phenological pattern and the sensitivity of time-varying feature selection in an iterative analysis process. In a casual overview process, we recognize a special phenological pattern around the Menifee City in southern California. When zoomed to the area centered at Menifee at a fine granularity level, A near-circular area exhibits a very unique pattern because of its short time period of "being green" when visualizing the animated rendering of the static features. The area of such pattern mostly matches with the freeform brushing in Figure 6.15a. After the interactive brushing of the interested

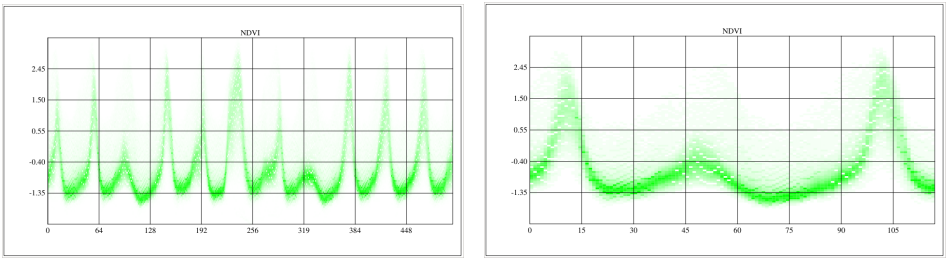
area, the composited time-varying features highlighted is rendered and displayed in Figure 6.15c. From the 11-year phenological pattern, we noticed that the yearly patterns of 2002 and 2007 has a much lower peak "greenness" measurement. Hence, we create a multi-year time-varying feature template (Figure 6.15d) that before and after 2002 and use it to select features over the whole CONUS. The similarity distribution is showed in Figure 6.15b. The "top similar" selected features reside in Southern California, Arizona and Utah.

The user interaction of brushing target spatial locations is flexible to use but a different shape of brushing make difference at the input time-varying features to generate the template. In a iterative manner, users might need to refine their brushing upon the visual feedback from the Cloudscape system. Or even more, users can sketch based on existing template of rendering of highlighted features. The later approach is demonstrated in Section 6.4.2 instead. Figure 6.15 contains another set of figures about the interactive brushing, feature template and shows two different sets of freeform brushing, feature template and similarity distribution.

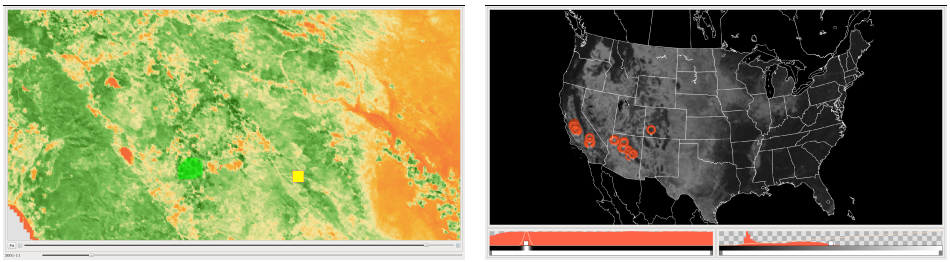
The first one is the circular brushing around the city mentioned above and it contains patterns with more example time-varying features and the second is a compact brushing restricted to a smaller but still highly representative region. The composited highlighted features as well as the feature template created exhibits very similar but clear looks with the first selection process, which means the first feature template is more uncertain while the latter one is more accurate. However the difference is not very obvious and the selection result does not vary much either in spatial similarity distribution. From the comparison of similarity score histogram, we can also tell that the average similarity score of the second feature template is lower and the percentage of the higher matched time-varying features are also less than the first one.



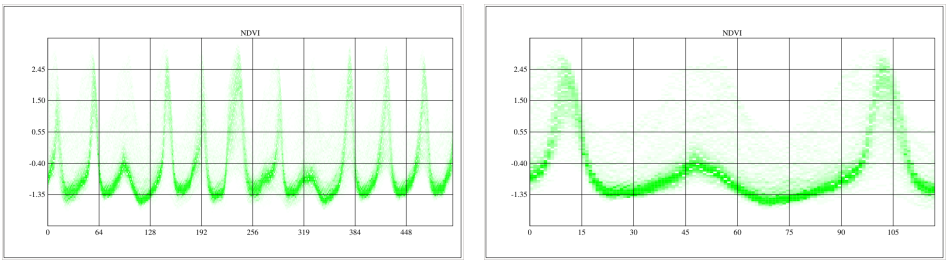
(a) Freeform brushing **(b)** Similarity distribution



(c) Phenological pattern (11 years) **(d)** Feature template (2.5 years)



(e) Freeform brusing **(f)** Similarity distribution



(g) Phenological pattern (11 years) **(h)** Feature template (2.5 years)

Figure 6.15: A multi-year time-varying feature template is created based on Menifee, California and its neighboring regions between Jan. 2001 to Jul. 2003 (107 time steps). The highlighted area exhibit both a unique yearly and a inter-annual phenological pattern.

6.4 Power Grid Situation Awareness

Analyzing the historical frequency and voltage characteristics using power grid measurement data set, FNET, domain experts are able to identify the normal and abnormal grid operational status. Our collaborators especially care about the extrema as well as the temporal events that is widely spread and could lead to severe malfunctioning of the power grid.

Same with the feature analysis application on the phenology data set, domain experts using the CO_3 Inspector and Cloudscape system often start their feature selection process from their possessed knowledge on certain type of time-varying features or sub-regions in the spatio-temporal domain. In our analysis application on the FNET data set, the possessed domain knowledge include a known large-scale frequency oscillation event that is called "storm" period and a type of voltage variation pattern named as FIDVR (Fault Induced Delayed Voltage Recovery) events. The "storm" event was recorded on April 27, 2011 over the whole East Interconnect due to the functional failures of two power stations. The event lasts about half a minute from 21:22:10 to 21:22:39 (UTC time); the FIDVR events is the phenomenon whereby system voltage remains at significantly reduced levels for several seconds after a transmission, subtransmission, or distribution fault has been cleared (NERC, 2009).

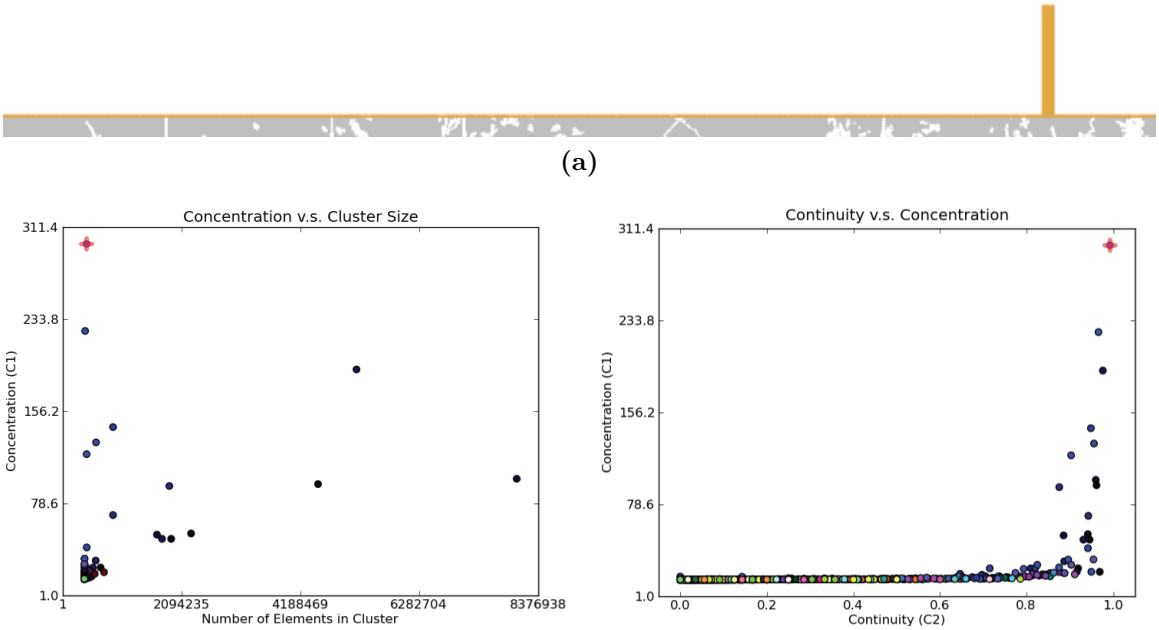
In this work, the raw FNET data set are abstracted into static features in two ways - hierarchical clustering on every 1-second events and discretely quantized and indexed outputs from the raw variable values (voltage and frequency).

6.4.1 Selecting Significant Features Using CO_3 Metrics

Example 1:

In the FNET data set, a small number of features standing out from the majority in the metric space represent possibilities of discovery. A tiny, highly concentrated and

continuous feature (in the top left of Figure 6.16b and the top right of Figure 6.16c) proves to be unique after further study. The corresponding operation state dominantly appears across most of the Eastern Interconnect but only within a very short period after the two generators' temporary failure (shown by the solitary tall bar in the temporal histogram). This feature has an exceptionally low frequency but a large phase angle shift. Its dominance indicates that this state is characteristic of the gradual process of the grid recovering to normal operation. Further study of this would assist in understanding the recovery process and help technicians respond quickly to severe power grid failures.



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

Figure 6.16: An example from the FNET data showing a highly concentrated and continuous feature, with an exceptionally low frequency but a large phase angle shift. Temporal histogram (a) shows this feature is exclusive to a small window of time after the "Storm Period". (1s bin size)

Example 2:

With the FNET data set, we can analyze the uncommon event called a "Storm Period" by computing the co-occurrence graph composed solely of the features that occurred

during that time (Figure 6.17). The co-occurrence graph layout clearly reveals 6 characteristic groups that provide additional information upon further examination. Figure 6.17(a) reveals that one group (highlighted in red) involves features with both high frequency variation and high voltage variation and occurred most frequently during the first half of the day (UTC time 00:43:12 to 08:52:48). Figure 6.17(b) shows another feature group that had a heavier presence in the second half of the day (UTC time 12:00:00 to 21:07:12) and exhibited smaller variations in frequency but larger ones in voltage. As visualized by the parallel coordinate rendering in Figure 6.17(e) *vs.* Figure 6.17(f), the contrasting behavior could help to motivate further domain science research to explain the cause and progression of a "Storm Period".

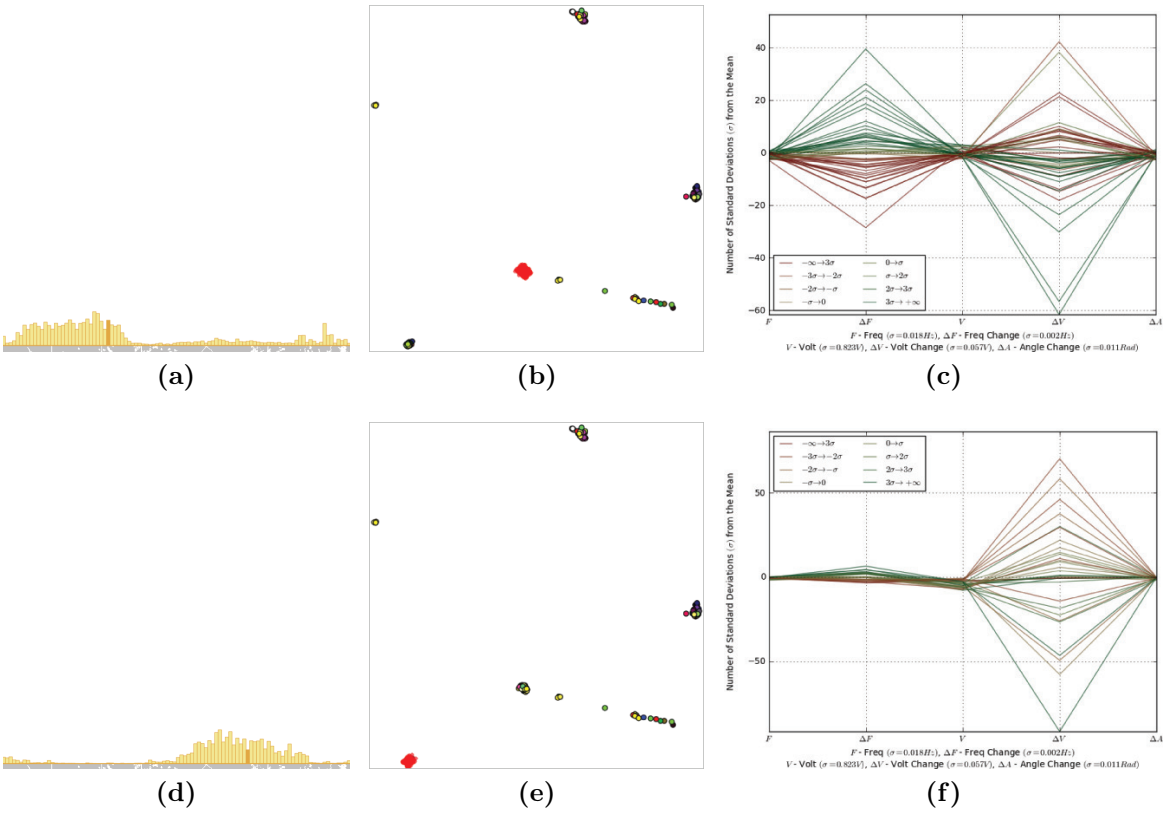


Figure 6.17: Two example feature groups in the FNET data; both occurred in a "Storm Period". Both are inherently correlated in terms of co-occurrence, but contain highly contrasting distribution patterns. (1s bin size)

6.4.2 Selecting Significant Time-varying Features Using Cloudscape

Similar to Cloudscape for MODIS, Cloudscape for FNET has also enabled the domain users conduct a feature selection in a near-interactive rate and study the similarity distribution in temporal space for each of the locations whose feature sets are used for time-varying feature selection. The section has mainly focused on demonstrating the use of the Cloudscape system in selecting both transient and lasting features of both the frequency and voltage characteristics.

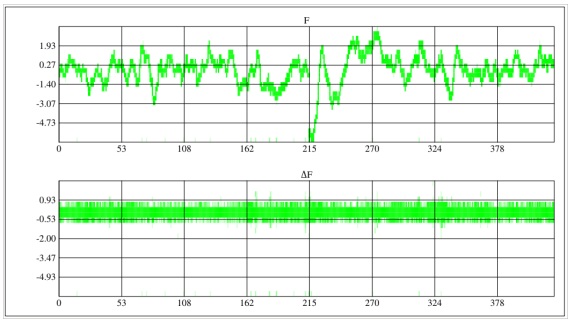
Different from our CO_3 metrics study of this dissertation work that the static features abstracted from the FNET data sets are the clusters on 1-second events, the time-varying features all has a varied length which is determined by the users or by the time-varying feature alignment.

Example 1

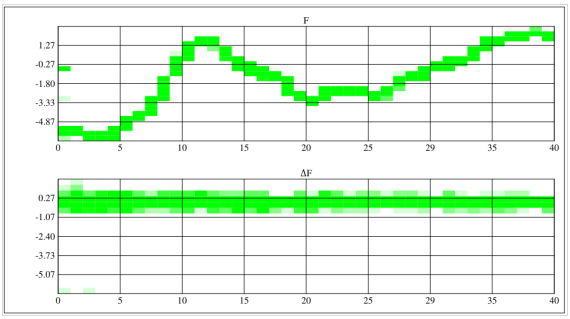
Also studied in CO_3 metrics' application in selecting static features, the "storm" period happened on April 24, 2011 is also a very interesting analysis target in time-varying feature selection. Users can interactively visualize the whole day's frequency variation and it is not difficult for us to recognize a frequency recovering stage shortly after the "storm" period (starting from 21:36:12 UTC time) when the frequency on the grid recovers from particularly low (below -6.4σ) to the normal status.

Figure 6.18a shows an overall view of the frequency variation at a coarse level of the time period between 20:00:00 and 23:00:00 (UTC time) on April 27th, 2011. We take the frequency recovery period of time at same granularity level and create time-varying feature template from it. The users are still allowed to generate feature templates from group of features, however, unlike the MODIS phenology data set that the phenological patterns has a much more stable seasonal variation and a group of spatially co-located features are more likely to be similar with each, the FNET power grid measurement data records highly dynamic frequency and voltage characteristics

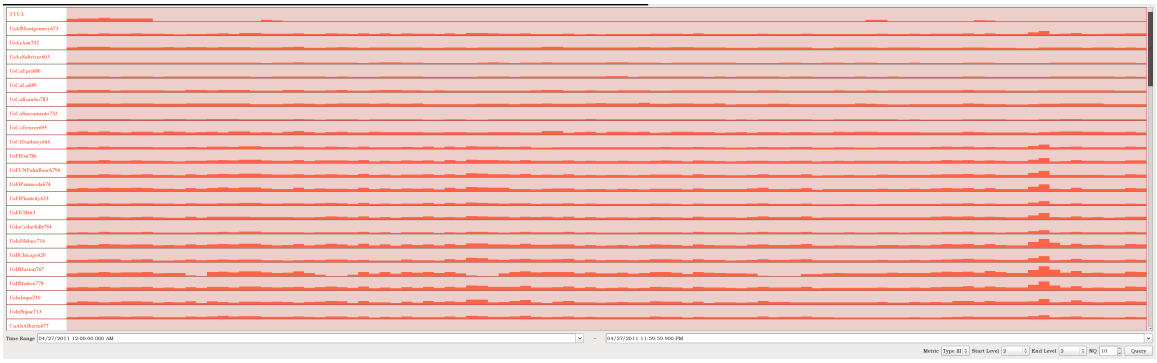
at a very fast rate. As a result, grouping a set of features is not as meaningful as that in the phonology feature selection process. The generated time-varying features template is showed in Figure 6.18b. We then apply the feature selection using the template at all the locations on April 27th and the temporal similarity distribtuion is rendered and compared row by row. Out of no surprise, many of the location shows a high match during the same period of time. The "storm" event has been prevalent over the whole Eastern Interconnect but does not affect the other interconnects. The fact is confirmed with the fact that the sensor locations out of Eastern Interconnect (the ones in California, Arizon and etc.) does not return any similar frequency pattern while the ones within Eastern Interconnect does (Figure 6.18c).



(a) Overall view



(b) Feature template



(c) Similarity distribution of all locations

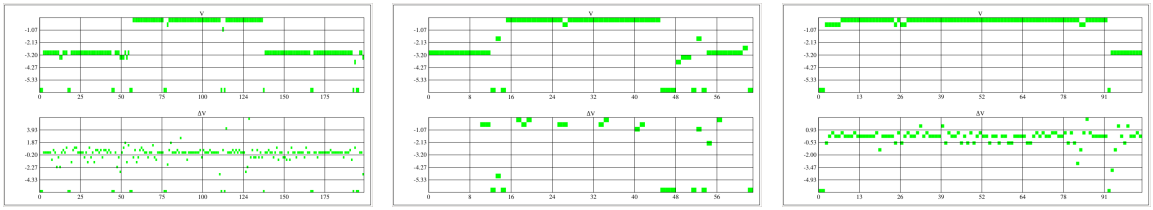
Figure 6.18: A time-varying feature template is created based on the frequency variation pattern between 21:36:12 and 21:52:51 at the sensor location UsTnUtk692.

Example 2

This example is used to illustrate the feature selection process of a typical transient time-varying feature. Figure 6.19a is the representative time-varying feature template created from a transient step-shaped voltage variation. In about 8 seconds, the system voltage of the power grid sharply changed between two voltage levels ($-3\sigma - 0\sigma$). and the whole feature template lasts for 20 seconds (200 time steps). Compared with the daily resolution of 864,000 time steps, such transient features are only a very small piece in the whole data set. Selecting significant features based on such feature templates is especially difficult due to the extremely large search space. The multi-scale search strategy can help to speed up the feature selection process, however, the users should always select a proper starting granularity to avoid losing too much necessary informations for an accurate selection of time-varying features. For a demonstration purpose, we pick three locations and compare the feature selection results in the whole 9-month time span one by one with each of their temporal similarity histogram are expanded in size as showed in Figure 6.19b. The height of each histogram bar is interpreted as the maximum selection similarity score within a certain period of time. For the 9-month timespan, each bar out of the 100 represents about 3-day period of time. These locations include one (CaMbHydroc719) in Manibota, Canana and two locations in Florida, United States (UsFLNPalmBeach796 and UsFlUfl663). Among the three locations, we can visually tell that CaMbHydroc719 and UsFLNPalmBeach796 contains a few best matches of which their similarity scores are obviously larger than the rest. Contrastly, UsFlUfl663 shows up a weaker feature selection result. Each of the best matched time-varying features are also showed in the figure.

(a) Feature template

(b) Comparison of similarity distribution



(c) Top selected feature in Us-FLNPalmBeach796, 13:00:00 - 13:00:20, 2011-07-17, similarity score = 1.0

(d) Top selected feature in CaMb-Hydroc719, 16:53:12 - 16:53:19,2011-07-20, similarity score = 0.542714

(e) Top selected feature in UsFlUfl663, 18:16:34 - 18:16:44, 2011-04-14, similarity score = 0.472362

Figure 6.19: A time-varying feature template is created using a recorded transient step-shaped voltage variation pattern. The feature selection result using the created template of three locations including UsFLNPalmBeach796, CaMbHydroc719 and UsFlUfl663 are compared on the similarity distribution.

Example 3

In this section, we have taken the FIDVR event for example illustration of the iterative sketch-based method of creating time-varying feature templates. We started from a pre-detected FIDVR event starting from 16:38:15 to 16:38:30 (UTC time) that recorded by the sensor UsMoFranklin756 on July 9, 2011. In the first stage of the iterative feature selection, we directly use the 15-second FIDVR voltage variation pattern is used to create the template. The template is then used to select similar time-varying features, potential FIDVR events as well, in the 9-month voltage record of the same sensor. The top three selected features (the example feature self-included) and similarity distribution is showed in the first column of Figure 6.20.

For a different way, we would like to reduce the complexity of the feature template and focus solely on the voltage recovery part in which the voltage has kept a trend of increasing till backing to normal Figure 6.20b is the generated feature template.

and the top two selected features are within roughly the same time period while the third one changed due to the selection template. And the average similarity score is much lower than the first template as the majority of most-occurred static features are removed from the template creation and would no longer contribute the feature alignment process.

Additional to the first two created templates, we further refine the time-varying feature template through a sketched-based method that applys directly in the variable space. Using a semi-transparent paint brush, we extend the second FIDVR feature template to a fuzzier variation pattern by adding some static features with lower weights to each time step which increase the possibility of selecting FIDVR-like time-varying features. The change in the template has affect the selection result in that the average similarity score dropped to a new level and the a different set of top similar time-varying features are selected.

The sketch-based refine is particularly useful when the domain users are very familiar with what variation pattern they would like to select. It could a small tweak over an existing feature template and could also be complete new from scratch.

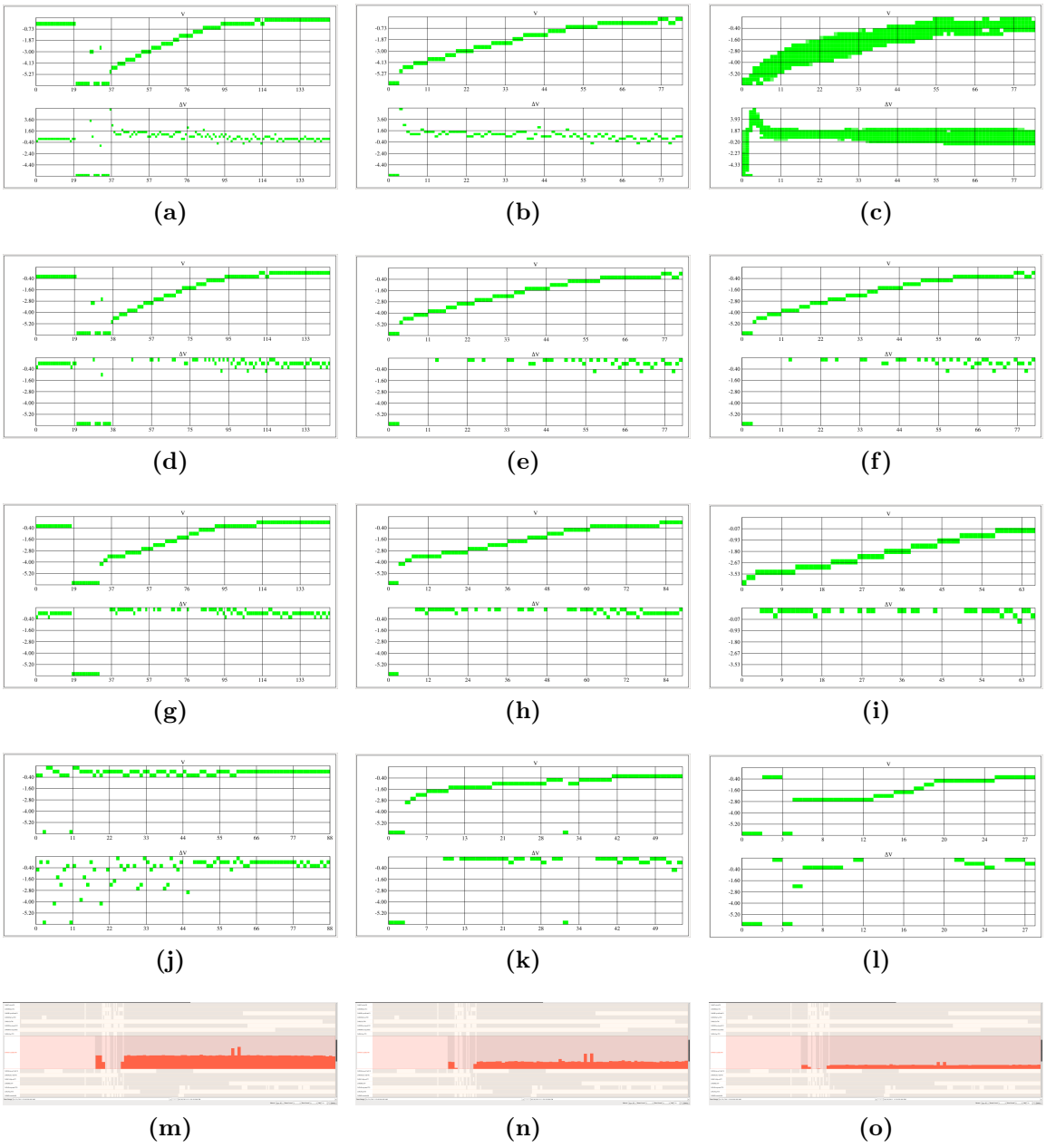


Figure 6.20: Three time-varing feature templates created based on a detected FIDVR voltage variation pattern are created consecutively in an iterative feature selection process. From left to right, each column shows the feature template, top three selected time-varying features and over temporal similarity distribution.

6.5 Discussion

With respect to both static and time-varying feature selection practices for both of our application data sets, navigating through the hierarchical feature space containing multiple levels of granularities is particularly difficult for domain users since the total number of features is beyond a person's ability for the traditional click-and-view analysis process. The CO_3 Inspector with backend CO_3 metrics evaluation and the Cloudscape system that enables an interactive time-varying feature selection within a large data set represent a new possible solution to facilitate visual and interactive feature analysis and selection from two aspects:

1. Assist domain users to better define their analysis interest. The proposed approaches include developing either quantitative metrics (CO_3) as well as example-based feature generalization (Cloudscape).

2. Speedup the feature selection process. The approaches include pre-preprocessing that reduces the runtime complexity and ManyView system framework that further exploits the computation resources.

The application examples in this section illustrate our capability to effectively select significant features or group of features, both static and time-varying, demonstrate the power of our proposed methodological methods in exposing previously unknown possibilities to users. Such feature selection capability we demonstrate is crucial as data sets consistently and quickly grow in size and complexity. Our domain experts from climate modeling and power systems find CO_3 metrics and the Inspector system to be useful for analyzing historical data. They were intrigued by the feature groups discovered by the Inspect system, and expressed a perception of a high level of utility and future potential.

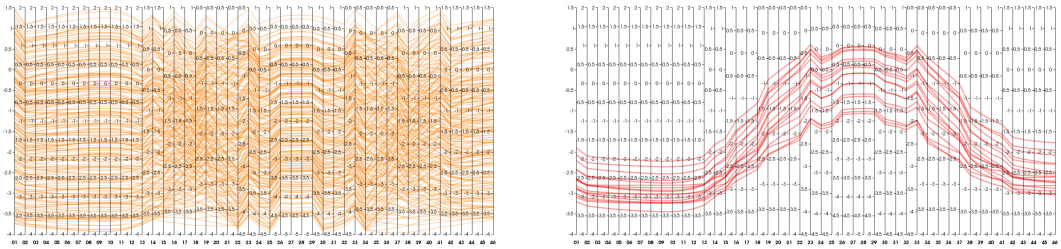
With respect to CO_3 metrics, they are especially useful to summarize physical space properties of features extracted from attribute space. Our use of coarse grained bins is general and novel for handling high resolutioned spatial and temporal data

sets. In a different way, the Cloudscape system has made the interactive visualization and analysis of features in large-scale data set possible on commodity computing resources - an ability previously not reported in literature, by a series of architectural and methodological components.

Compared with the selecting approach of multivariate or time-varying features using range query method, both of the CO_3 metrics and the Cloudscape system are flexible to users to select more informative features.

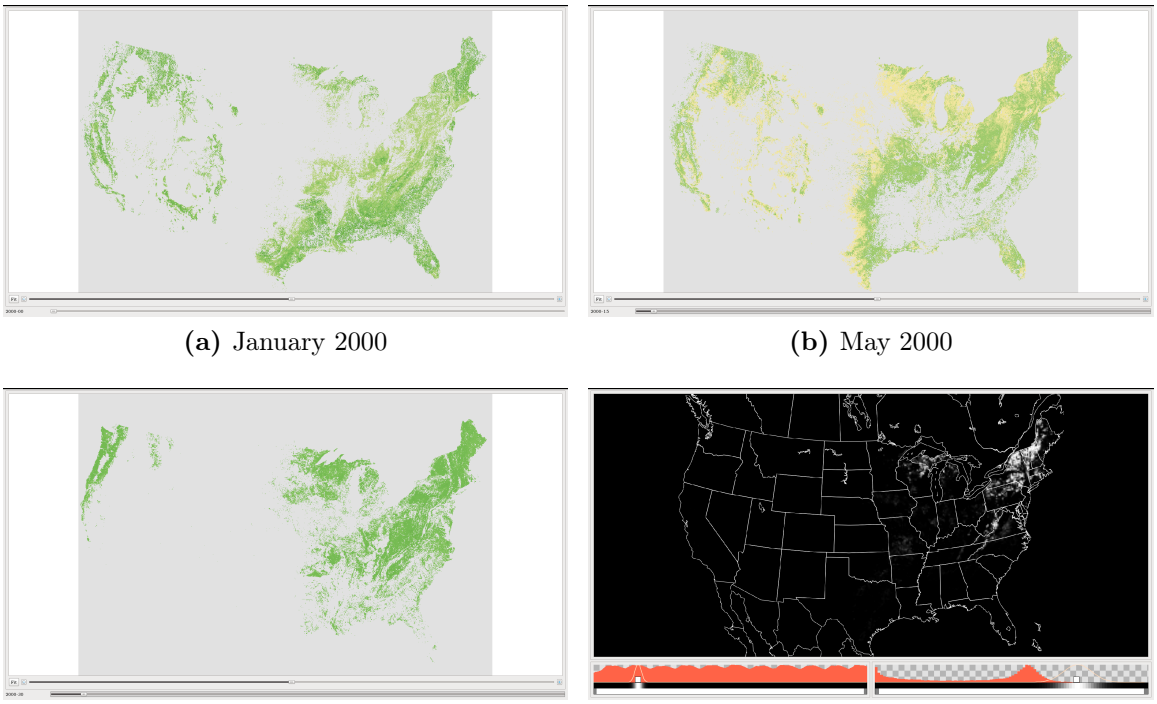
Take the static feature selection using CO_3 metrics for example, using parallel coordinates, users are able to specify ranges for one or multiple variables to select a subset of all the features. Figure 6.21b is a parallel coordinates plot corresponding to the first selected feature group shown in Figure 6.7. All features in the same attribute space are plotted in orange and shown in Figure 6.21a. Here, the selected features share similar variation patterns while the actual values of the vegetation indices are not particularly close to each other and therefor not possible to be directly selected.

For a time-varying feature, the concept of range query can be generalized to setting a set of discrete static features rather than a range of variable values. Figure 6.22 shows the distributions of static features existed in Adirondack Mountain areas at three different time steps as well as the distribution of "top similar" features selected using Cloudscape system. Each of the them is the result to a range query at a single time step and presents a different pattern from each other and also the selection using Cloudscape. Unless each of the time step is strictly set to the same with the target time-varying feature, the results would contain more features than necessary because of those features only match with the target feature at the set time steps. Even if the target time-varying feature is fully described in all the time steps, the range query method is unable to select the features that are similar but start or ends at a different time step or has a slightly different variation pattern and also provides no information about the degree of similarity in selection.



(a) **(b)**

Figure 6.21: Traditional parallel coordinates plots of example feature group showed in Figure 6.7 (rendered with the other features in the same attribute space in different colors)



(a) January 2000 **(b)** May 2000

(c) September 2000 **(d)** Top similar features selected using Cloudscape

Figure 6.22: Range-queried static features sharing the same "greenness" condition with Adirondack Mountains area at three different time steps in 2000. Each of them presents a different pattern.

Chapter 7

Conclusion

In this dissertation, I have mainly addressed the challenges of effective selection of static and time-varying features in large-scale observational data sets, targeting at the domain users with only access to commodity computing and analysis resources, a community that is greatly emerging but is largely neglected by the state-of-art scientific researches. Our solution to the problem is a complete framework comprised of 1) a hierarchical clustering tool to extract features of over 3 billion multivariate data elements; 2) a series of quantitative metrics, CO_3, that summarizes the physical space characteristics of extracted features to evaluate and select meaning static features; 3) an example-based feature generalization method to describe a feature selection target upon possessed knowledge; 4) a system that efficiently and thoroughly manages the computing and analysis resources. The framework has been demonstrated through two representative situation awareness applications, both of them have not only a large size but also particularly large spatio-temporal resolutions. The major accomplishments of this dissertation work is that we have proposed a general framework for analyzing large observational data and improved the interaction of feature selection, especially for time-varying features, to a level that users are able to effectively select features from a real data set exceeding 500GB in size on a single

SMP workstation as demonstrated. In other words, the users can expect to easily adapt our framework to a similar observational data set at a near-terabyte scale.

The tools, methods, and system that developed in this dissertation work are a few first steps of us to solve a scientific problem that is challenging now and will be in the near future, as the complexity of data sets, including size, number of variables, data quality, and etc., continues increasing.

The work has a few limitations, and they can be improved in several aspects. First, a few parts of the work requires non-trivial parallel preprocessing. Next, our method would not offer a different benefit over previous methods, if the data set is relatively manageable or the feature set is already well understood.

Another future research opportunity refers to adapting the whole framework to the cloud computing architecture so that the domain users without access to supercomputers can also analyze their problems at a much larger scale using readily more computation resources from the cloud. Nevertheless, the architecture difference leads to a different set of challenges that require further studies.

Bibliography

Bahmani, B., Moseley, B., Vattani, A., Kumar, R., and Vassilvitskii, S. (2012). Scalable k-means++. *Proc. VLDB Endow.*, 5(7):622–633. 12

Cantú-Paz, E., Newsam, S., and Kamath, C. (2004). Feature selection in scientific applications. In *Proc. tenth ACM SIGKDD Int. Conf. Knowl. Discov. data Min.*, KDD '04, pages 788–793, New York, NY, USA. ACM. 10

Childs, H., Pugmire, D., Ahern, S., Whitlock, B., Howison, M., Weber, G. H., and Bethel, E. W. (2010). Extreme Scaling of Production Visualization Software on Diverse Architectures. *IEEE Comput. Graph. Appl.*, 30(3):22–31. 15, 40, 44

Cleland, E. E., Chuine, I., Menzel, A., Mooney, H. a., and Schwartz, M. D. (2007). Shifting plant phenology in response to global change. *Trends Ecol. Evol.*, 22(7):357–65. 6

Connor, M. and Kumar, P. (2009). Fast construction of k-nearest neighbor graphs for point clouds. *IEEE Trans. Vis. Comput. Graph.*, 16(4):599–608. 20

Crawl, D. and Altintas, I. (2008). A Provenance-Based Fault Tolerance Mechanism for Scientific Workflows. In Freire, J., Koop, D., and Moreau, L., editors, *Proven. Annot. Data Process.*, volume 5272 of *Lecture Notes in Computer Science*, pages 152–159. Springer Berlin Heidelberg. 16

Doerr, K. and Kuester, F. (2011). CGLX: A Scalable, High-Performance Visualization Framework for Networked Display Environments. *Vis. Comput. Graph. IEEE Trans.*, 17(3):320–332. 16, 40

Doleisch, H., Mayer, M., Gasser, M., Priesching, P., and Hauser, H. (2005). Interactive Feature Specification for Simulation Data on Time-Varying Grids. *SimVis*, pages 291–304. 11

Eilemann, S., Makhinya, M., and Pajarola, R. (2009). Equalizer: A Scalable Parallel Rendering Framework. *Vis. Comput. Graph. IEEE Trans.*, 15(3):436–452. 16, 40

Endsley, M. R. (1995). Toward a Theory of Situation Awareness in Dynamic Systems. *Hum. Factors J. Hum. Factors Ergon. Soc.*, 37(1):32–64. 2

Ferreira Cordeiro, R. L., Traina Junior, C., Machado Traina, A. J., López, J., Kang, U., and Faloutsos, C. (2011). Clustering very large multi-dimensional datasets with MapReduce. In *Proc. 17th ACM SIGKDD Int. Conf. Knowl. Discov. data Min.*, KDD '11, pages 690–698, New York, NY, USA. ACM. 12, 13

Glatter, M., Huang, J., Ahern, S., Daniel, J., and Lu, A. (2008). Visualizing Temporal Patterns in Large Multivariate Data using Modified Globbing. *IEEE Trans. Vis. Comput. Graph.*, 14(6):1467–1474. 9, 11

Hadwiger, M., Laura, F., Rezk-Salama, C., Hollt, T., Geier, G., and Pabel, T. (2008). Interactive Volume Exploration for Feature Detection and Quantification in Industrial CT Data. *IEEE Trans. Vis. Comput. Graph.*, 14(6):1507–1514. 9

Hargrove, W. W. and Hoffman, F. M. (2004). Potential of multivariate quantitative methods for delineation and visualization of ecoregions. *Environ. Manage.*, 34 Suppl 1:S39–60. 13

Ingram, S., Munzner, T., Irvine, V., Tory, M., Bergner, S., MoÌĹller, T., and Möller, T. (2010). DimStiller: Workflows for dimensional analysis and reduction. In *Vis. Anal. Sci. Technol. (VAST), 2010 IEEE Symp.*, pages 3–10. 58

Justice, C. O., Vermote, E., Townshend, J. R. G., Defries, R., Roy, D. P., Hall, D. K., Salomonson, V. V., Privette, J. L., Riggs, G., Strahler, A., Lucht, W., Myneni,

R. B., Knyazikhin, Y., Running, S. W., Nemani, R. R., Wan, Z., Huete, A. R., van Leeuwen, W., Wolfe, R. E., Giglio, L., Muller, J., Lewis, P., and Barnsley, M. J. (1998). The Moderate Resolution Imaging Spectroradiometer (MODIS): land remote sensing for global change research. *Geosci. Remote Sensing, IEEE Trans.*, 36(4):1228–1249. 6

Kendall, W., Jian Huang, Peterka, T., Latham, R., and Ross, R. (2011a). Toward a General I/O Layer for Parallel-Visualization Applications. *IEEE Comput. Graph. Appl.*, 31(6):6–10. 15, 44, 47

Kendall, W., Wang, J., Allen, M., Peterka, T., Huang, J., and Erickson, D. (2011b). Simplified parallel domain traversal. In *Proc. 2011 Int. Conf. High Perform. Comput. Networking, Storage Anal.*, SC '11, pages 10:1———10:11, New York, NY, USA. ACM. 13

Keogh, E. and Lin, J. (2005). Clustering of time-series subsequences is meaningless: implications for previous and future research. *Knowl. Inf. Syst.*, 8(2):154–177. 10

Keogh, E. and Ratanamahatana, C. A. (2004). Exact indexing of dynamic time warping. *Knowl. Inf. Syst.*, 7(3):358–386. 34

Kincaid, R. (2010). SignalLens: Focus+Context Applied to Electronic Time Series. *Vis. Comput. Graph. IEEE Trans.*, 16(6):900–907. 12

Kohl, J. A., Wilde, T., and Bernholdt, D. E. (2006). Cumulvs: Interacting with High-Performance Scientific Simulations, for Visualization, Steering and Fault Tolerance. *Int. J. High Perform. Comput. Appl.*, 20(2):255–285. 16

Kola, G., Kosar, T., and Livny, M. (2004). A fully automated fault-tolerant system for distributed video processing and off-site replication. In *Proc. 14th Int. Work. Netw. Oper. Syst. Support Digit. audio video*, NOSSDAV '04, pages 122–126, New York, NY, USA. ACM. 16

Kumar, J., Mills, R. T., Hoffman, F. M., and Hargrove, W. W. (2011a). Cluster Analysis-Based Approaches for Geospatiotemporal Data Mining of Massive Data Sets for Identification of Forest Threats. *Procedia CS*, 4(0):1612–1621. 6, 9, 13, 66, 70

Kumar, J., Mills, R. T., Hoffman, F. M., and Hargrove, W. W. (2011b). Parallel k-Means Clustering for Quantitative Ecoregion Delineation Using Large Data Sets. *Procedia Comput. Sci.*, 4(0):1602–1611. 12, 13

Le Moal, G., Moraru, G., Veron, P., Rabate, P., and Douilly, M. (2012). Feature selection for complex systems monitoring: An application using data fusion. In *Commun. Comput. Control Appl. (CCCA), 2012 2nd Int. Conf.*, pages 1–6. 11

Li, K., Hibbs, M., Wallace, G., and Troyanskaya, O. (2005). Dynamic scalable visualization for collaborative scientific applications. In *Parallel Distrib. Process. Symp. 2005. Proceedings. 19th IEEE Int.*, page 6 pp. 16

Lin, J., Keogh, E., Lonardi, S., Lankford, J. P., and Nystrom, D. M. (2004). VizTree: a tool for visually mining and monitoring massive time series databases. In *Proc. Thirtieth Int. Conf. Very large data bases - Vol. 30*, VLDB '04, pages 1269–1272. VLDB Endowment. 12

Liu, G., Zhang, H., and Wong, L. (2012). Finding minimum representative pattern sets. In *Proc. 18th ACM SIGKDD Int. Conf. Knowl. Discov. data Min.*, KDD '12, pages 51–59, New York, NY, USA. ACM. 9

Liu, H. and Yu, L. (2005). Toward integrating feature selection algorithms for classification and clustering. *Knowl. Data Eng. IEEE Trans.*, 17(4):491–502. 10

Liu, J., Chen, Y., and Zhuang, Y. (2013). Hierarchical I/O Scheduling for Collective I/O. *2013 13th IEEE/ACM Int. Symp. Clust. Cloud, Grid Comput.*, pages 211–218. 16

Maciejewski, R., Jang, Y., Woo, I., Janicke, H., Gaither, K., and Ebert, D. (2012). Abstracting Attribute Space for Transfer Function Exploration and Design. *IEEE Trans. Vis. Comput. Graph.*, PP(99):1. 9

Mansour, E., Allam, A., Skiadopoulos, S., and Kalnis, P. (2011). ERA: efficient serial and parallel suffix tree construction for very long strings. *Proc. VLDB Endow.*, 5(1):49–60. 13

Markham, P. N., Liu, Y., Bilke, T., and Bertagnolli, D. (2011). Analysis of frequency extrema in the eastern and western interconnections. In *Power Energy Soc. Gen. Meet. 2011 IEEE*, pages 1–8. 14

Martin, S., Brown, W. M., Klavans, R., and Boyack, K. W. (2011). OpenOrd: an open-source toolbox for large graph layout. In *Proc. SPIE 7868, Vis. Data Anal.* 60

Mei, K., Rovnyak, S. M., and Ong, C.-M. (2008). Clustering-Based Dynamic Event Location Using Wide-Area Phasor Measurements. *Power Syst. IEEE Trans.*, 23(2):673–679. 14

Messina, A. R., Vittal, V., Ruiz-Vega, D., and Enriquez-Harper, G. (2006). Interpretation and Visualization of Wide-Area PMU Measurements Using Hilbert Analysis. *Power Syst. IEEE Trans.*, 21(4):1763–1771. 14

Mitra, P., Murthy, C. A., and Pal, S. K. (2002). Unsupervised Feature Selection Using Feature Similarity. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24:301–312. 10

Möller-Levet, C., Klawonn, F., Cho, K.-H., and Wolkenhauer, O. (2003). Fuzzy Clustering of Short Time-Series and Unevenly Distributed Sampling Points. In R. Berthold, M., Lenz, H.-J., Bradley, E., Kruse, R., and Borgelt, C., editors, *Adv. Intell. Data Anal. V*, volume 2810 of *Lecture Notes in Computer Science*, pages 330–340. Springer Berlin Heidelberg. 10

Mulder, J. D., van Wijk, J. J., and van Liere, R. (1999). A survey of computational steering environments. *Futur. Gener. Comput. Syst.*, 15(1):119–129. 16

Müller, M. (2007). Dynamic Time Warping. In *Inf. Retr. Music Motion*, pages 69–84. Springer Berlin Heidelberg. 33

NERC (2009). A Technical Reference Paper Fault-Induced Delayed Voltage Recovery. 82

Overbye, T., Klump, R., and Weber, J. (2003). Interactive 3D Visualization of Power System Information. *Electr. Power Components Syst.*, 31(12):1205–1215. 14

Patwary, M. A., Palsetia, D., Agrawal, A., Liao, W.-k., Manne, F., and Choudhary, A. (2012). A new scalable parallel DBSCAN algorithm using the disjoint-set data structure. In *Proc. Int. Conf. High Perform. Comput. Networking, Storage Anal.*, SC '12, pages 62:1—-62:11, Los Alamitos, CA, USA. IEEE Computer Society Press. 12

Post, F. H., Vrolijk, B., Hauser, H., Laramee, R. S., and Doleisch, H. (2003). The State of the Art in Flow Visualisation: Feature Extraction and Tracking. *Comput. Graph. Forum*, 22(4):775–792. 9

Pretorius, A. J. and Van Wijkk, J. J. (2006). Visual Analysis of Multivariate State Transition Graphs. *Vis. Comput. Graph. IEEE Trans.*, 12(5):685–692. 11

Qiu, B., Chen, L., Centeno, V., Dong, X., and Liu, Y. (2001). Internet based frequency monitoring network (FNET). In *Power Eng. Soc. Winter Meet. 2001. IEEE*, volume 3, pages 1166 –1171 vol.3. 7

Rakthanmanon, T., Campana, B., Mueen, A., Batista, G., Westover, B., Zhu, Q., Zakaria, J., and Keogh, E. (2012). Searching and mining trillions of time series subsequences under dynamic time warping. In *Proc. 18th ACM SIGKDD Int. Conf. Knowl. Discov. data Min.*, KDD '12, pages 262–270, New York, NY, USA. ACM. 13

Ross, R., Carns, P., and Metheny, D. (2010). Parallel File Systems. In Chan, Y., Talburt, J., and Talley, T. M., editors, *Data Eng.*, volume 132 of *International Series in Operations Research & Management Science*, chapter 8, pages 143–168. Springer US, Boston, MA. 15

Singh, B., Sharma, N. K., Tiwari, A. N., Verma, K. S., and Singh, S. N. (2011). Applications of PMUs in an Integrated Power System Environment with FACTS Controllers: A Survey. *Int. J. Eng. Sci. Technol.*, 3(3). 14

Sips, M., Kothur, P., Unger, A., Hege, H.-C., and Dransch, D. (2012). A Visual Analytics Approach to Multiscale Exploration of Environmental Time Series. *Vis. Comput. Graph. IEEE Trans.*, 18(12):2899–2907. 12

Tzeng, F.-Y. and Ma, K.-L. (2004). A Cluster-Space Visual Interface for Arbitrary Dimensional Classification of Volume Data. In *Proc. Viss.*, pages 17–24. 9

Wan, Z., Wang, P., and Li, X. (2004). Using MODIS Land Surface Temperature and Normalized Difference Vegetation Index products for monitoring drought in the southern Great Plains, USA. *Int. J. Remote Sens.*, 25(1):61–72. 6

Wang, J., Sisneros, R., and Huang, J. (2013). Interactive Selection of Multivariate Features in Large Spatiotemporal Data. In *IEEE Pacific Vis. Symp.* 10

Weber, M., Alexa, M., and Müller, W. (2001). Visualizing Time-Series on Spirals. In *IEEE Symp. Inf. Vis.*, page 7. IEEE Computer Society. 11

Wong, P. C., Schneider, K., Mackey, P., Foote, H., Chin, G., Guttromson, R., and Thomas, J. (2009). A Novel Visualization Technique for Electric Power Grid Analytics. *Vis. Comput. Graph. IEEE Trans.*, 15(3):410–423. 14

Woo, I., Maciejewski, R., Gaither, K. P., and Ebert, D. S. (2012). Feature-Driven Data Exploration for Volumetric Rendering. *IEEE Trans. Vis. Comput. Graph.*, 18(10):1731–1743. 9

Yin, J., Zheng, Z., and Cao, L. (2012). USpan: an efficient algorithm for mining high utility sequential patterns. In *Proc. 18th ACM SIGKDD Int. Conf. Knowl. Discov. data Min.*, KDD '12, pages 660–668, New York, NY, USA. ACM. 9

Zhang, Y., Markham, P., Xia, T., Chen, L., Ye, Y., Wu, Z., Yuan, Z., Wang, L., Bank, J., Burgett, J., Conners, R. W., and Liu, Y. (2010). Wide-Area Frequency Monitoring Network (FNET) Architecture and Applications. *IEEE Trans. Smart Grid*, 1(2):159–167. 14

Zhong, Z., Xu, C., Billian, B. J., Zhang, L., Tsai, S.-J., Conners, R. W., Centeno, V. A., Phadke, A. G., and Liu, Y. (2005). Power system frequency monitoring network (FNET) implementation. *IEEE Trans. Power Syst.*, 20(4):1914–1921. 7

Vita

Jingyuan Wang was born in Quanzhou, Fujian, China on May 8, 1989. He graduated from Yongchun No. 1 Middle School in July 2005, and received his Bacherlor's degree in Electrical Engineering at Nanjing University in July 2009. In August 2009, he began his Ph.D study at the University of Tennessee at Knoxville and completed with the Doctor of Philosophy degree in Computer Engineering in December 2014.