Query-Driven Visual Explorations of Large-Scale Raster Geospatial Data in Personal and Multimodal Parallel Computing Environments

By Jianting Zhang @CCNY

Project Description

1 Introduction

Advances in geospatial technologies, especially space-borne satellite remote sensing and large-scale environmental modeling, have generated large amounts of raster geospatial data at the ever increasing spatial and temporal resolutions. High resolution large-scale raster geospatial datasets provide tremendous opportunities to understand the Earth and our environments deeper than ever before. Unfortunately, existing data management techniques on large-scale raster geospatial data mostly focus on storage, standard-based remote access and simple or animated display. Very little interactive query functionality is available to users to stimulus hypothesis and subsequently to seek causal relationships. As a result, scientists are usually forced to manually inspect images converted from large-scale raw data for exploratory analysis. This may bring quite some potentially serious problems: 1) Using down-scaled data to reduce workloads may make high-resolution data meaningless in enhancing scientific discovery. 2) Subsetting large-scale data to limit geographical scopes may miss large-scale patterns, especially global patterns. (3) Using predefined data-to-image conversion methods through quantization may reduce data accuracies and generate false patterns.

Query-Driven Visual Explorations (QDVE) integrates dynamic queries with traditional visualization techniques to support scientists to explore large-scale datasets interactively [1-7]. As QDVE applications are both data intensive and computing intensive, most of the reported works utilize cluster computers in supercomputing centers equipped with large numbers of processors, fast network connections and sophisticated parallel I/O software. While the barriers to utilize supercomputing resources have been significantly lowered over the past few years due to the increasingly community efforts in developing cyberinfrastructure tools [8,9], we believe that QDVE applications are more suitable to personal computing environments due to its interactive exploration nature. This is especially desirable to general researchers and citizen scientists that do not have privileges to conveniently reserve supercomputing resources for their exclusive uses. As commodity personal workstations and desktop computers are increasingly equipped with larger numbers of multicore Chip Multiprocessors (CMPs) [10][11]and may-core General Purpose Graphics Processing Units (GPGPUs)[12][13], we envision the possibility and opportunities of developing and deploying QDVE applications on personal computers.

While a current commodity personal computer with dual hex-core CPUs and two GPU cards can easily achieve computing power at the teraflops level, which largely removes computing bottleneck in QDVE applications, a more fundamentally difficult problem is the disk I/O bottleneck. Compared with cluster computers that usually have dedicated I/O control hardware and sophisticated parallel I/O software over fast network connections, disk I/Os on personal computers are serial and are limited to around 100MB/s or even lower. The mismatch between disk I/Os and processors has put significant technical challenges in developing QDVE applications in a personal computing environment. We list a few use cases to help understand the problem domain and the relevant queries in QDVE applications. While these queries can be answered through offline preprocessing followed by standalone visualization of the results, we argue that the uninterrupted exploration processes promised by QDVE techniques are likely to facilitate novel scientific discoveries more effectively.

1) WorldClim current (1950-2000) climate data [14][15] has been widely used in climate change and biodiversity research. The data includes altitude, monthly precipitation, minimum temperature and maximum temperature (12 month) and 18 derived bioclimatic variables. At the global 30 arc-seconds spatial resolution, each of the 55 raster datasets has 43200*21600 raster cells. Ecologists, and biogeographers are interested in knowing regions with January temperature between [t1,t2) and July precipitation between [p1,p2) as they may be suitable for the distributions of species X. Our experiments showed that reading the January precipitation raster dataset (1.5 GB) from disk to memory will cost 15-18 seconds on a 7200rpm disk using a Dell T5400 workstation. However, the data loading time can be reduced to 2.37 seconds if the raster is divided into multiple 2048*2048 tiles and use the 16 threads available to the dual quad-core machine to read and uncompress the compressed tiles.

2) Quite a few indices from satellite data, such as Normalized Difference Vegetation Index (NDVI), Normalized Difference Water Index (NWDI) and Normalized Difference Drought Index (NDDI), have been derived and used as indicators of geographical or environmental phenomena, events or processes [16]. The NASA Moderate Resolution Imaging Spectroradiometer (MODIS) 500 meter spatial resolution data has a grid size of 31200 by 21600 in North and South Americas. With 417 time steps (every 8 days) and seven bands for each scene at a time step, the data amounts to 1.14 TB in nine years from 2000 to 2009. Experiments reported in [7] showed that it took around 1 minute to identify regions that satisfy a simple compound range query on four variables for draught assessment using 16K CPU cores on a Jaguar Cray XT4 supercomputer. Based on the experiment results, it is clear that the read time dominants the whole QDVE pipeline (46.56 seconds out of 57.68 seconds), even if sophisticated parallel I/O components have been used.

3) The National Mosaic & Multi-Sensor Quantitative Precipitation Estimation (NMQ) data [17] covers the continental United States with a spatial resolution of 1 kilometer and temporal resolution of 5 minutes. This translates to a data volume of more than 10GB raw data per day and nearly 4TB per year. Climatologists and hydrologists may want to examine the differences between National Weather Service (NWS) in-situ observation records and the NMQ data in a W*W window around NWS station locations. For those deviations that are larger then threshold T, they may further want to explore the casual relationships on how the deviations vary with topography (e.g., altitude, slope and aspect) at a regional scale (e.g., within a W'*W' window) using a local regression approach.

2 Vision, Goals and Overview of the Research Plan

Raster data representation is a major data model for geospatial data [18]. Surprisingly, compared to vector geospatial data that hundreds of tree indexing techniques have been developed [19,20], raster geospatial data is much less well supported in databases with respect to efficient query processing. Existing techniques in spatial databases (e.g., Oracle GeoRaster [21] and RasDaMan [22]) adopt a chunking approach to store raster geospatial data and index the metadata (e.g., minimum and maximum) of the chunks using standard vector spatial indexing. While queries on the spatial locations and metadata values of the chunks are supported, chunks are stored as Binary Large Objects (BLOBs) and usually no queries and other operations on the chunks are supported. We are not aware of any major efforts in systematically support efficient I/O in querying large-scale raster data beyond the chunk level in spatial databases. On the other hand, while some scientific data formats (such as HDF5 [23,24]) supports data compression at the chunk level, their focuses are on data storage, simple display and animated visualization and are not query-friendly. Queries are usually considered as part of a modeling process and require offline processing. However, mixing well-structured and frequently used queries with physics-based modeling often makes it complicated to reuse the query processing component, especially when modeling systems are evolving from serial to parallel computing.

To effectively support QDVE of large-scale raster data and foster new scientific discoveries in a personal computing environment, we propose to extend existing spatial databases functionality by making full use of available commodity parallel hardware resources on personal computers. We consider the following technical issues relevant to seeking a scalable and high-performance solution for query driven visual explorations of large-scale raster data.

- The data-intensive nature of query-driven visual explorations requires novel data management techniques. Queries generally are required to be answered within a few seconds in order to be interactive. The time may be even insufficient to load all relevant data into memory from hard drives. The problem becomes severe as the general trends are that data resolution and volume are going up while the ratios between I/O speeds and processor speeds on current commodity personal computers are going down.
- Supporting ad-hoc and low-response time queries simultaneously is likely to go beyond the capability of a single processor and thus parallel computing is essential to achieve the goals. Parallel data structures and algorithms that are applicable across different commodity hardware platforms, including multicore CPUs, GPGPU accelerators and cloud/grid computing clusters are crucial to seek scalable computing solutions. While it is difficult to automatically parallelize generic serial code, pilot studies on parallelizing certain domain-specific algorithms, such as raster geospatial data, can shed lights on seeking more general solutions. Due to the data-parallel nature of large-scale raster data, supporting QDVE-related queries can be an ideal pilot study.

- While indexing techniques have been widely applied in speeding up both relational and vector geospatial data query processing, the ad-hoc nature of visualization and visual explorations related queries very often makes it inadequate to use pre-generated indices targeted at repetitive queries involving fixed numbers of dimensions. Large volumes of disk-resident indices can potentially slow down query processing due to the disk I/O bottleneck. The balance between indexed and non-indexed data access techniques need to be re-evaluated in the context of visual explorations of large-scale raster geospatial data using new cost models on new parallel hardware architectures.

We consider the mismatch between slow, serial disk I/Os and fast, parallel processors (including both CPU and GPU cores) on modern commodity personal computers a key technical challenge for data-intensive applications. Our goals in this project are to develop a novel data-centric, software-oriented framework to mitigate such mismatch and a set of techniques to effectively support QDVE of large-scale raster geospatial data. While indexing has been the primary technique in reducing disk I/Os in processing relational queries in the past few decades, we argue that many of the increasingly sophisticated indexing techniques may not be suitable for QDVE queries in exploring large-scale numeric raster geospatial data. For example, many QDVE queries involve identifying regions-of-interests (ROIs) from multivariate geo-referenced environmental data where each variable is represented as a raster layer. Selection ratios in many ROI queries are usually high once certain levels of the multi-scale representations of rasters have reached. The low selectivity will incur extra disk I/Os to retrieve indices while the majority of the data items need to be fetched from disks regardless. Our proposed framework focuses on balancing top level indexing and lower level data encoding (compression/decompression) by fully utilizing the parallel computing power available to modern personal computers. The framework is realized by developing realistic cost models that incorporate both index guided queries and data decompression before fine tuning of key parameters. Based on the framework, a set of techniques will be developed to effectively support QDVE queries on large-scale raster geospatial data. While the primary focus of the proposed research is to speed up online query processing using a single personal workstation (or *Personal Computing*), the parallel-aware techniques are also suitable to guide indexing and compressing data in grid or cloud computing environments in an offline or semi-online mode for scalable data preprocessing and on-demand workload sharing as illustrated in Fig. 1. We term the hybrid computing environment as *Multimodal Computing*. The proposed research and development efforts are grouped into four tasks as listed below and each of the four tasks will be further detailed in Section 4.

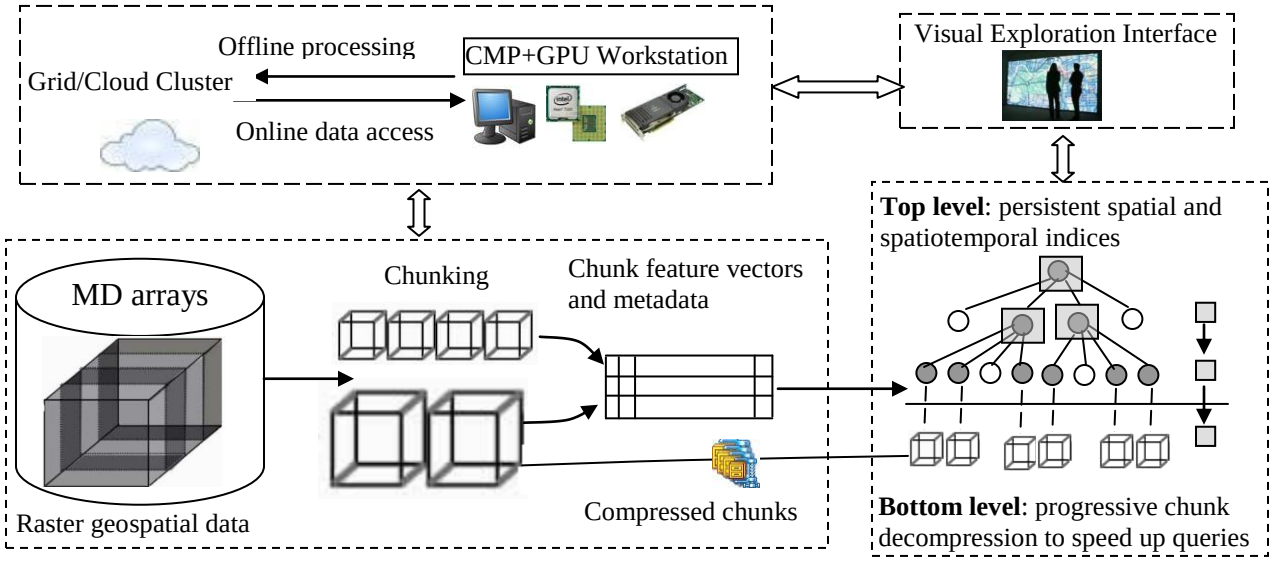

Fig. 1 Illustration of System Architecture: QDVE in Personal and Multimodal Computing Environments

(1) Understand the characteristics of major queries in QDVE applications in the context of visual explorations of large-scale raster geospatial and analyze existing data management practices on raster geospatial data in both spatial databases and Geographical Information System (GIS). A subset of geospatial operations that are suitable for QDVE applications will be identified.

(2) Develop realistic cost models to model end-to-end query performance of the identified QDVE queries for the Disk-CPU-GPU data transfer pipeline under both non-compression and data-compression scenarios. The cost models will serve as the foundation for fine-tuning chunk sizes and data layouts using both fixed parameters (e.g., disk to memory data transfer speed, CPU to GPU data transfer speed) and dynamic parameters (e.g., data compression ratios, decompression algorithm complexities).

(3) Develop parallel data structures and algorithms to facilitate parallel query processing on both multicore CPUs and GPGPU accelerators. A particular challenge we propose to address is to develop low-complexity algorithms with bounded and low memory footprints that can potentially run across multiple types of parallel hardware.

(4) Develop a semi-automated task scheduling module to coordinate local (CPU and GPU) and remote (grid and cloud) cluster computing resources for both online query processing and offline data preprocessing.

Intellectual Merit:

The proposed framework enables query driven visual explorations of large-scale raster geospatial data on commodity personal computers equipped with multicore CPUs and GPU accelerators. Compared with cluster computing based solutions, our proposed software-driven approach is more affordable, accessible and usable to general users. The framework can be used as a replacement of traditional cluster computing for moderate sized problems as well as can serve as a "last-mile" sub-system to be connected to large cyberinfrastructure for extremely large size problems by dividing QDVE tasks among personal workstations and cluster computers. While using data compression to reduce data storage and transfer costs in the context of relational queries and image coding have been studied extensively, to the best of our knowledge, we believe the idea on tightly integrating spatial indexing and raster data compression for processing QDVE queries on parallel hardware attached to personal computers is novel and innovative. The proposed research bridges the gaps between scientific data management, digital image processing, parallel computing, spatial databases and environmental applications in the context of visual explorations of large-scale raster geospatial data. The uninterrupted exploration processes promised by QDVE techniques are likely to facilitate novel scientific discoveries more effectively

Broader Impact:

The proposed research and prototype development have the potential to turn an individual researcher's desktop computer into high-performance visual exploration tools at no extra cost and thus have a broad audience in the geospatial data processing communities. The visual exploration tools are particularly interesting to researchers whose work are related to remote sensing applications and environmental modeling, such as atmospheric science, oceanography, hydrology, environmental monitoring, biodiversity, global changes and emergency responses. The prototype system can be used to exemplify how a high-performance parallel GIS can be realized (at least partially) on a commodity personal computer and serve as the "last-mile" sub-system for larger cyberinfrastructure at the same time. This is likely to foster community efforts on utilizing parallel computing for more general geospatial analysis.

3 Background and Related work

Advancements in remote sensing technology and instrumentation have generated huge amounts of remotely sensed imagery from air- and space-borne sensors. In recent years, numerous remote sensing platforms for Earth observation with increasing spatial, temporal and spectral resolutions have been deployed by NASA, NOAA and the private sector. The next generation geostationary weather satellite GOES-R (whose first lunch is scheduled in 2015) will improve the current generation weather satellite by 3, 4 and 5 times with respect to spectral, spatial and temporal resolutions [25]. With a temporal resolution of 5 minutes, GOES-R generates 288 global coverages everyday for each of its 16 bands. At a spatial resolution of 2km, each coverage and band combination has 360*60 cells in width and 180*60 cells in height, i.e., nearly a quarter of a billion cells. In addition to raw imagery data, derived data products targeting at domain-specific applications are fast growing as well. For example, regional and global coverage of Land Cover, Land Surface Temperature, Albedo and Vegetation Indices are among a fraction of MODIS Product Table [26]. Still yet, numerous environmental models, such as Weather Research and Forecast (WRF[27]), have generated even larger volumes of geo-referenced raster model

output data with different combinations of parameters, in addition to increasing spatial and temporal resolutions.

Traditionally environmental scientists are limited to simple display and animation of raster geospatial data in a desktop environment. Very little high-performance visual exploration functionality has been provided despite the continuous community software development efforts, e.g.[28-30]. While large-scale raster geospatial data are widely used for environmental modeling in a variety of domains in different regions, existing software tools provide very limited query capability as observed by Jim Gray and his research group [31]. Most raster geospatial data are numeric and can have a large number of distinct values. Since human eyes can only effectively distinguish a limited number of colors at a time, traditionally thematic mapping techniques have been applied to quantify the distinct values into bins and color the data using the bins (which has been provided by many GIS software). While a viable solution for small-scale datasets that can be rendered in a single screen and relatively well-perceived by domain experts, it is often too complex to choose suitable quantification methods and to interpret the rich information in large-scale environmental datasets. This is true even if a large tiled display wall is setup to display all the cells as pixels. Research in Information Visualization has shown that human eyes can only effectively distinguish a limited number of colors at a time. Screen resolutions beyond 4000 by 4000 pixels are not effective [32].

Previous studies have shown that query-driven visual exploration [1-3,5,7,33] is a valuable approach to deep understanding of scientific data. These studies couple high-performance query processing techniques (such as FastBit [34,35] and M-ary search tree [3]) and visualization techniques (such as histograms, Parallel Coordinate Plot-PCP and iso-surfaces). QDVE techniques have been successfully applied to multiple domains, such as astrophysics [3], biogeochemistry [36], combustion simulation [4,37], climate [2,4,5], Computational Flow Dynamics (CFD) [3,4], laser wakefield simulation [5] and traffic analysis [38]. While several of these pioneering QDVE works have considered time as a special dimension, spatial dimensions were not explicitly addressed. Indeed, FastBit and M-ary search tree indexing data structures utilized in these applications require treating a multivariate record as a multi-dimensional point and indexing was performed at the individual record level. As such, geospatial dimensions were treated the same as thematic dimensions and no distinctions between domains and ranges can be made for geospatial analysis. A problem in applying existing QDVE techniques to geospatial raster data is that the links between geographic distributions (domains) and thematic values (ranges) have to be established through enumerating record identifiers, i.e., raster cell identifiers, which is likely to incur scalability issues as further discussed below. We also note that many of the previous works focused on 3D rendering through iso-surfaces constructions. While adding a third dimension certainly provides richer information, the number of raster cells along the spatial (X/Y) dimensions are generally limited to a few hundreds to a few thousands although grid-based cluster computing resources have already been extensively used. Our proposed research has a 2D geospatial focus and targets at 10-100 times higher resolutions along each of the 2D geospatial dimensions. In addition, our targeted computing platforms are personal workstations although we plan to use on-demand cluster computing resources for “surge protection”.

Raster geospatial data can be abstracted as a special type of multi-dimensional array data with at least longitude (X) and longitude (Y) dimensions and optional time (T), elevation (Z), spectral and other semantic dimensions. Different from vector geospatial data that has the notion of object identifiers which may be associated with rich semantics, raster cells, as the basic units of raster geospatial data, need to be aggregated and put into context before they become useful for further analysis. We consider QDVE serves as the first steps towards deep understanding the data and develop fine-tuned algorithms to extract vector features from raster data if necessary. However, the primary goal of QDVE techniques is to fast identify subspaces in a multi-dimensional data space that are potentially useful for further investigation. QDVE can be considered as an intermediate step between a simple raw raster representation of complex real world and a more precise representation using a vector data model. As such, QDVE techniques are distinct to both traditional display/animation based visualization (where the underlying data space is treated uniformly) and feature extraction using sophisticated pattern recognition techniques. Experiments on aggregating Fastbit query records (multi-dimensional point) into regions of interest show that the aggregation stage overheads can dominate the QDVE process [39]. In contrast, separating domain dimensions and range dimensions and allow efficient mapping between these two categories of dimensions using hierarchical data structures can support queries with rich semantics. For example, as detailed in Section 4, instead of supporting compound range queries only as in most of

existing QDVE applications, our proposed research supports at least four types of queries. In this sense, we consider our domain-driven solution a valid contribution to the development of QDVE techniques.

Parallel and distributed processing of geospatial data has been evolving with mainstream hardware architectures, including shared disk and shared nothing systems. Quite a few works on parallel spatial data structures [40-42], spatial join [43,44], spatial clustering [45], spatial statistics [46,47] , polygon vectorization [48][49] and handling terrain data [50] have been reported. However, as discussed in [51], research on parallel and distributed processing of geospatial data prior to 2003 has very little impact on mainstream geospatial data processing applications, possibly due to the accessibility of hardware and infrastructures in the past. Cary et al [52] reported their experiences on processing spatial data with MapReduce on a Google and IBM cluster using the Hadoop framework. Their experiments include R-Tree construction on point data and image tile quality computation. Parallel computing on LIDAR data using cluster computers [53] is getting increasingly popular due to its computation intensive nature. More recently, works reported [54,55] have demonstrated significant speedups by using cluster computing for spatial statistics. Among these works, very few of them specifically addressed query processing of large-scale raster data, especially in an online, interactive, visual exploration context. We have observed that there are significant gaps between new hardware architectures and software information systems that can fully utilize the hardware features. While both multicore CPUs and GPGPUs have been available for a few years and have been explored in many other application domains, to the best of our knowledge, neither GIS nor spatial databases (open source or commercial) can fully utilize the hardware potentials (our review in [56] provides more discussions and references).

Sophisticated visualization techniques and systems have been proposed to visualize spatiotemporal and/or multivariate data [32,57-79] and some of them have been applied to scientific data [33,80-83]. However, most of them assume that all data can be fit into main-memory and thus efficient query processing of large-scale data does not play a substantial role in these techniques. The recently emerging Visual Analytics initiative [68,84,85] advocates for tightly coupling visualization and analytics techniques to effectively explore large-scale data. Different from QDVE applications that originate from scientific computing community and use massive parallel cluster computers to process large-scale data from the very beginning, many of the existing research on visual analytics rely on personal workstations to provide high degree interactions that are difficult to achieve through remote rendering at the clusters. Our proposed research is thus related to both QDVE and visual analytics, which integrates the advantages from both through careful redesigns of data structures, algorithms and system architectures in a personal computing environment. We believe our efforts in developing query processing techniques that trade off computing with I/O for large-scale raster data through data compression can potentially stimulate similar research on exploring domain-specific data semantics and push the limit of visual analytics of large-scale data on personal workstations.

We want to point out that compressing data to reduce I/O is not a completely new idea. Lossy and lossless data compression techniques have been routinely used to reduce storage and transmission overheads for geospatial data, including large-scale raster geospatial data. In addition to using generic compression techniques, patented commercial systems, such as MrSID from LizardTec [86] and Wavelet-based ECW from EARDAS/ERMapper [87], have been quite influential in geospatial data processing communities. However, these products are not designed to support queries and the close-source nature makes it very difficult to extend them to support QDVE applications. It is also unclear whether the techniques behind the products can be migrated to multicore CPUs and/or many-core GPUs. In databases, while quite a few works have investigated the role of data compression in reducing I/O overheads and direct query processing on compressed data [35,89,90], our proposed research targets at large-scale raster geospatial data which is quite different from relational data. Furthermore, existing works are prior to the wide availability of current generation of commodity personal computers equipped with multicore and GPGPU accelerators. The changed hardware architectures and cost models may require a different set of data compression techniques and query processing algorithms. We also note that, data compression has also been explored in cluster computing environments [91,92] as data compression can be an integral function of dedicated hardware I/O controllers. However, most of existing compression algorithms applied to cluster I/O controllers are “semantic unaware”, i.e., data are simply treated as bit streams for lossless compression and decompression. No data semantics can be explored to facilitate efficient query processing. We consider that trading off computing with I/O in a personal computing environment is technically more challenging than in a cluster computing environment as both available hardware I/O bandwidth and computing power are much more limited. As such, lossless data

compression and decompression can hardly achieve the desired degree of tradeoff and make it practically useful. Different from existing lossless data compression techniques used in cluster computing environments, our proposed research focuses on simultaneous data compression and filtering based query processing.

Wavelet [93,94] based image compression and transmission over different types of communication channels have been extensively studied in digital image processing community [95-99]. However, we argue that Wavelet compression in multimedia applications are mostly designed for viewing grayscale or RGB color images and may not be suitable for querying purposes, especially for large-scale raster data. While indexing and query processing techniques have been developed for image databases [100], they are mostly applied to derived multi-dimensional features after image decompression rather than on raw image data. Although many wavelet based image compression techniques (including wavelet transformation and coefficient encoding stages [99]) can be extended to geospatial raster data with higher numbers of bits per cell (pixel), our proposed research tightly integrates spatial indexing with both wavelet transformation and encoding to support efficient queries on large-scale raster geospatial data. In contrast, previous databases applications of wavelet transformations focus on histogram approximation [98], range aggregate queries in OLAP [102-104] and approximate relational queries in the wavelet domain [105]. As these techniques are applicable to multidimensional arrays, they can be tailored to raster geospatial data. However, while choosing wavelet synopses [106,107], i.e., a subset of significant wavelet coefficients, can potentially reduce disk I/O overheads, the computation overheads of the proposed approaches are generally much higher than well-established wavelet encoding schemes for digital image processing, e.g., bit-plane based ones. Jahangiri et al [108] provided two novel operations to establish relationships among wavelet coefficients. These techniques are not suitable for our application context, i.e., speeding up serial I/O through parallel compression/decompression. Finally we note that, while parallel algorithms for wavelet transformation are emerging [109-112], practical software-based wavelet coefficient encoding and decoding solutions largely remain serial.

Query processing on multicore CPUs [113-115], GPGPU accelerators [116-119] and cluster computing facilities [120-124] have gained considerable research interests in the past few years as multicore CPU becomes the mainstream architecture and commercially available cluster computing resources. Equally important, GPGPU technologies have turned massive floating-point computational power of a modern GPU's graphics-specific pipeline into general-purpose computing power. According to [13], NVIDIA alone has shipped almost 220 million CUDA [125] based GPGPU devices. The number is several orders of magnitude more than historical massively parallel architectures. Due to the ever-increasing demands from massive game players, the computing power as well as benefit-cost ratios of GPGPU devices will improve significantly and steadily in the near future. In fact, it is projected that GPU speed improvement will be more than 50% per year, over the next decade [126]. While hardware architectural changes require significant changes on data structures and algorithms and system re-implementations in the context of geospatial data processing, we believe that the "growing pain" will help the research community better understand the unique and rich parallelism among geospatial data and geospatial operations, and, develop fully-fledged parallel GIS and spatial databases by fully utilizing parallel hardware both locally (multicore CPUs and GPGPUs) and remotely (cloud/grid computing).

4 Research Plan

As discussed before, our goals are to develop a novel data-centric, software-based framework to mitigate the mismatch between serial disk I/Os and parallel processors attached to personal computers and a set of techniques to effectively support QDVE applications in both a personal and a multimodal computing environment.

4.1 Understanding large-scale raster data and QDVE Queries

Advances in geospatial data processing have identified various types of modeling and query techniques [127,128]. While it is difficult to clearly separate between modeling and query, in general, queries are more well-defined with known computation complexities, general-purposed and can be processed in a short time. Continuous improvements of computing capabilities have moved traditional offline modeling tasks to online query processing. However, in general, only queries whose inputs can be conveniently defined from graphics user interface and whose outputs can be mapped to some visualization primitives are suitable for visual exploration purposes. Towards this end, we propose to re-examine various types of raster geospatial operations, including local, focal, zonal and global operations

[128], and evaluate their suitability for QDVE applications. We propose a practical approach by studying the categorization of geoprocessing tools available from ESRI ArcGIS software due to its large user community [56]. For each of the geoprocessing tools that are related to raster data operations, we will first evaluate their suitability for QDVE applications and investigate whether they are suitable for multicore CPU or GPGPU implementations within the project group and then conduct online surveys in different user communities that may have different focuses.

While our priority list on targeted queries types for visual explorations of large-scale raster geospatial data may change dynamically, the following query types are on our initial list: 1) **Extended Spatial Window (ESW) query** to allow users limit spatial extent. Different from window/range queries for vector spatial data [129] where a set of object identifiers are returned, a ESW query on raster data can return internal representation (e.g., quadtrees) of raw data chunks that fall within the spatial window together with metadata, including predefined and dynamically generated statistics. 2) **Region of Interests (ROI) query** [130] to allow user to limit value ranges and locate corresponding spatial distributions of raster cells that satisfy the range criteria. ROI queries can be used separately on whole datasets or combined with ESW queries to further limit spatial extents based on range values. A complex ROI query may be composed from multiple simple ROI queries that involve only one variable using any conjunctive or disjunctive combinations. ROI queries can be considered as extensions to the data manipulation components of several traditional visualization techniques, such as linking and brushing. 3) **Optimized Raster Algebra (ORA) query**. While traditional map/raster algebra operations [127] work at individual raster cell level and are considered as local operations, the metadata/statistics associated with the raster data chunks can be used to efficiently prune query space with support from domain knowledge. For example, assuming that a1<x<b1 and a2<y<b2 where x and y represent two raster variables, then c=(x-y)/(x+y) will be in the range of c1=(a1-b2)/(b1+b2) and c2=(b1-a2)/(a1+a2). Further assume the query range value is [q1,q2), then any chunk whose c value range [c1,c2) does not intersect with [q1,q2) can be safely pruned. 4) **Local Spatial Statistics (LSS) query**. Local spatial statistics, e.g., Spatial Autoregression (SAR) [128] and Weighted Geographical Regression (GWR) [131], can be used to show how the statistics vary across space and have played an important role in spatial data analysis [132]. LSS queries for interactive visual explorations allow users to interactively define resolution level, spatial extent and important parameters (such as GWR bandwidth) and visualize the query results using proper forms, e.g., 3D topographical surface [133]. LSS queries are usually computationally intensive and are ideal for demonstrating the utilization of local GPUs and remote grid/cloud computing resources.

4.2 Cost Models: Investigating the Role of Data Compression

Fig. 2 depicts a simplified system model of a typical personal workstation equipped with multicore CPUs and GPGPUs. Limited by both the hard dive bus interface and its controller interface, commodity hard drives usually have an effective I/O speed in the range of 50-150 MB/s [10,134]. On the other hand, the current generation PCI-E interface between a CPU and a GPU allows a maximum of 4 GB/s one-way transfer rate [12]. It is clear that hard drive disk I/O speed is the primary bottleneck affecting the overall system throughputs in data intensive applications, including QDVE of large-scale raster geospatial data. Let the effective data transfer rates between hard drive and CPU and between CPU and GPU are $R_{D\text{-}C}$ and $R_{C\text{-}G}$, respectively. Let the raw data volume be V and the compression ratio be C. Further assume N_c and N_g parallel processing units on CPUs and GPUs are used to decompress the compressed chunks and the maximum decompression time durations for the units are T_{max}^C and T_{max}^G, respectively. Based on the simplified system model, we can see that the data transfer time from hard drive to CPU memory without compression is $V/R_{D\text{-}C}$ and the time with compression is $V/(C^* R_{D\text{-}C})+T_{max}^C$. If we choose to use GPU for decompression, then the total time to transfer the raw data to GPU (assuming single GPU for simplicity) is $V/(C^* R_{D\text{-}C})+V/(C^* R_{C\text{-}G})+ T_{max}^G$. If the decompressed data needs to be transferred back to CPU for further processing, the total time becomes $V/(C^* R_{D\text{-}C})+ V/(C^* R_{C\text{-}G})+T_{max}^G +V/R_{C\text{-}G}$. We note that both T_{max}^C and T_{max}^G can be a non-linear function of N_c and N_g, respectively.

Despite simple, the system model shows the major components in loading large-scale raster data from hard drives to CPU (and GPU) memories and highlights the important role of data compression. Unlike many traditional data and image compression techniques [135] that mostly target at reducing data volumes and increase compression ratios as much as possible, the new application requires optimizing decompression time which is jointly determined by compressed data volume and complexity of decompression algorithm. Thus, the model may favor low-complexity decompression algorithms (smaller decompression time) at the cost of larger compressed data volumes (larger I/O time) to reduce overall

data loading time. Second, the simplified model also suggests that whether to use GPU accelerator for chunk decompression depends on whether the benefits of using additional GPU computing power can surpass the CPU-GPU data transfer costs. If the decomposed data is used subsequently on GPU for further processing, the CPU to GPU data transfer cost could be justified. However, transferring large amount of decompressed data from GPU back to CPU is likely to overshadow the benefits. To develop realistic cost models for practical query optimizations in the application domain, we propose to address the following issues.

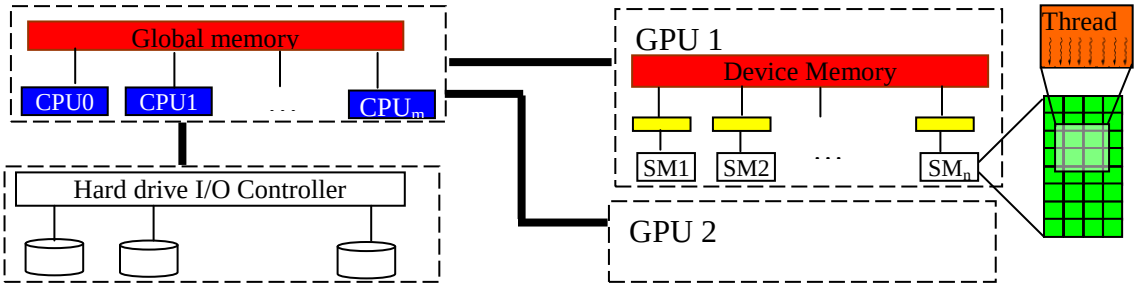

Fig. 2 A Simplified System Model for Personal Workstations with Multicore CPUs and GPGPUs

First, the distributions of raster cell values can be highly uneven and the data volumes of compressed chunks can vary significantly. The unevenness of decompressing chunks should be taken into consideration during the raster data chunking and compression stages. Second, assuming that each CPU core is assigned to process a single chunk at a time, the decompression times can vary significantly among all the processors which will subsequently affect T_{max}^{C}. Third, a large number of CPU processors may compete for shared resources (such as memory and last level cache) [114,136] and make it even harder to predict T_{max}^{C}. Finally, various factors may affect T_{max}^{G}, such as the layout of GPU computing blocks, memory access coalescing and using shared memory [12]. We propose to adopt a practical solution for the inherent intricate problem by deriving statistics from system logs, generating lookup tables for typical scenarios and using the lookup tables in respective cost models for online query optimizations. We project that partitioning CPU cores into an I/O group and a decompression group and using proper chunk sizes can be two effective strategies to improve overall data loading time. The projection is based on the heuristic that functionally partition CPU cores can potentially reduce I/O and memory contentions and the heuristic that proper chunk sizes can align chunks with hard drive caches and memory caches better. While the numbers of cores to be allocated to the I/O or decompression groups can be changed in applications, hard drive and memory caches on commodity personal computers are usually transparent to applications. The optimal parameters are better determined through experiments. We also want to stress that while cost model driven optimization can potentially generate a better query execution plan, when the computation overheads is deemed to be too big, we will switch to rule-based query optimization by utilizing a set of predefined rules. For example, only when decompressed data are subsequently used in GPUs for other computation should decompression be performed on GPUs.

4.3 Parallel QDVE query processing: data structures and algorithms

Distinct data structures and algorithms for two types of data in this research are required: trees at the top level and multi-dimensional arrays at the bottom level (Fig. 1). While there are quite a few spatial indexing [19,20] and data compression techniques [135] that can be potentially used to realize the proposed framework, we have chosen quadtree and wavelet, respectively. The most important factor behind the decision is that data structures and algorithms should run across multiple types of parallel hardware architectures, including local multicore CPUs and GPGPU accelerators and remote grid/cloud based cluster computers. Our proposed solution is based on the cache-conscious parallel quadtree construction algorithm that we have developed for GPGPUs [137]. We have found that the algorithm can be easily applied to multicore CPUs. Furthermore, as detailed next, the algorithm can also be used for parallel wavelet bit-plane coding on both GPUs and CPUs. Since the data structure uses an array representation of tree nodes, it can be loaded to local CPU and GPU memories and streamed to remote computing nodes without requiring serialization and de-serialization. In addition to being able to reuse the quadtree in both spatial indexing and wavelet coefficient coding, the block-based feature in wavelet transformation and coefficient coding also aligns with the chunking schema and quadtree indexing very

well as shown in Fig. 1. As discussed in Section 4.4, our preliminary results have shown that the quadtree data structure [137] can be combined and partitioned along both spatial and thematic dimensions, which is desirable in parallel tree indices constructions across multiple processing units.

As for wavelet-based techniques, they have been extensively studied in digital image processing community. While not designed to support queries on large-scale multivariate raster data, performance behaviors of wavelet-based data transformation, encoding, transmission and decoding on different computing architectures and hardware devices are well studied. Certain queries, such as range-sum queries, can be directly supported in the wavelet domain without requiring complete data decompression. The feature has been attractive to the database research community [104,105]. We plan to investigate the possibility of processing the four QDVE queries in the wavelet domain. Furthermore, computation in wavelet transformation stage is naturally suitable for parallel processing. Finally, many wavelet based approaches use bit-plane encoding to encode transformed wavelet coefficients. The feature makes it possible to improve QDVE query efficiencies through progressive query processing.

We next briefly discuss how the relevant data structures and algorithms can be integrated to effectively support the four types of QDVE queries identified in Section 4.1. For ESW queries, starting from the root, all quadrants need to be checked to determine whether they intersect with the query window and prune those who do not. The query is performed recursively until the leaf nodes of the index tree is reached. Following the chunk identifiers associated with the leaf nodes, the metadata of the chunks can be gathered and an optimal chunk loading schema is then calculated. The chunks will be loaded into parallel computing units and uncompressed (fully or partially) so that subsequent processing can be performed. For ROI queries, the recursive query process remains the same except that the decision on whether to go to a particular child quadrant is based on range values instead of spatial extents. When a ROI queries involve multiple raster layers, the chunks of all the layers under a quadrant representing a qualified leaf node may be needed to be made available simultaneously on a parallel processing unit. We will develop an efficient sequencing algorithm to assign the chunks to different parallel processing units by adopting a graph based approach and reuse the algorithms that we have developed for data broadcasting over wireless channels [138-140]. (3) For ORA queries, they can follow a similar procedure as ROI queries except that, rather than comparing query ranges against multiple ranges of variables to derive a boolean value directly, a new variable from multiple existing variables need to be derived before checking the values of the derived variable. While the difference is negligible when validating top level indices on CPUs from an implementation perspective, a stack is required to derive the values of the new variable for individual raster cells which can be both memory-intensive and computation-intensive on both CPUs and GPUs. We will develop approaches to efficiently evaluate algebraic expressions at the chunk level by effectively utilizing commodity parallel hardware features, e.g., fast shared memory on Nvidia GPUs [12]. (4) As a focal raster operation, parallel processing of LSS queries are relatively straightforward on regularly distributed raster data and mature tiling techniques can be applied to speed up the processing [12]. However, workload balancing becomes a serious issue when LSS queries are combined with ROI queries. The input of LSS queries, i.e., output of ROI queries, becomes irregular and/or clustered in this case. To support compositions of LSS and ROI queries, we propose to extend the idea that we have presented in [56] and develop efficient algorithms on both CPU and GPU as following. First, ROI query results for all chunks are indexed using quadtrees and the chunk-level quadtrees are combined with the original quadtree indices above the chunk level to form a complete quadtree. The large quadtree is then used to compute the computation intensity across the spatial extent of the raster data based on the mathematic expressions of the underlying spatial statistics. The quadtree enhanced with an expected computation intensity value with each quadrant will be used to find a data partition schema to partition raster cells into groups. As calculating a local statistics for a cell may involve data items in multiple groups, partial statistics for groups are calculated in parallel at the group level before they are summarized up to derive a total statistics for the cell.

From the above discussions we can see that top level spatial indexing is effective in pruning search space and reduce the workload for decompressing and evaluating chunks when processing ESW, AOI and ORA queries. To further reduce the workload on decompression, we have proposed a new raster data encoding approach that adopts wavelet transformation but uses our parallel cache-conscious quadtree [137] for bit-plane coding as illustrated in Fig. 3. While the technique may not generate the best compression ratio, the decoding is much less complex and can be parallelized. The parallelization can be achieved at the both bit-plane level when decoding bit streams and at the reduction stage to reconstruct wavelet coefficients. By using a pyramid data structure as shown in the right part of Fig. 3, decoding

based on the proposed approach completely eliminates the need for pointers and dynamic memory allocations which makes it suitable for both multicore CPU and GPGPU implementations. Furthermore, while the pyramid data structure may increase memory footprint by 1/3 of the volume of decoded data [137], since decoding can be done in-place, the memory footprint may be even smaller than some existing best approaches (e.g., SHPIT [96] EBCOT [97], SPECK [98] and LWT [99]). The low memory footprint makes it suitable to utilize CPU caches and GPU fast shared memory to further improve the performance of bit stream decoding. Note that the proposed quadtree bit-plane coding approach should work with any existing wavelet transformation algorithms/implementations including parallel ones on multicore CPUs and GPUs [109-111].

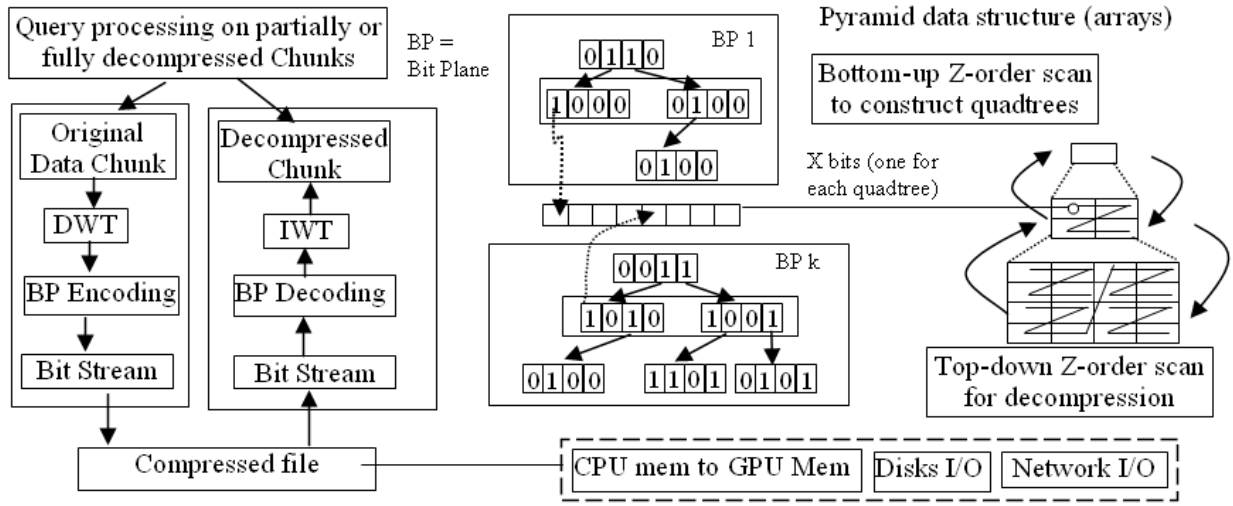

Fig. 3 Parallel Quadtrees for Wavelet Coefficients Coding Using a Pyramid Data Structure

Compared with other data compression algorithms, such as run-length and directory-based ones [135], one advantage of bit-plane encoding in the context of QDVE applications is that it can be used to supports progressive query processing. Assuming that the number of bit planes is N, then the contribution of the i^{th} bit plane to all raster cells vary between $[-2^i$ and $2^i]$. By incorporating the variations into inverse wavelet transformations, it is possible to filter out quadrants within the data chunk for AOI and ORA queries before the data chunk is fully decoded, i.e., early termination of query processing for unqualified quadrants. The similar idea can be used for progressive query processing where the accuracies are incrementally improved as more bit-planes are decoded. We consider both early termination and progressive evaluation in processing QDVE queries desirable features.

4.4 Extending to multimodal computing

We refer multimodal computing to integrating local personal computing with remote grid and cloud computing to enable visual explorations of large-scale raster geospatial data. The remote computing can be used for offline data preprocessing and aiding online query processing or even online indexing in certain cases. Furthermore, remote computing can be used to serve enhanced information in progressive query processing. Due to the data intensive nature of QDVE applications, shipping large amount of data to remote computing is better carried out prior to performing online QDVE tasks, i.e., during preprocessing stage. A scheduling module is essential to effectively coordinate local and remote computing. While considerable research from programming languages, operating systems and data networks have been carried out on task scheduling in parallel computing, it remains a fundamental difficult problem in seeking a generic, fully automatic solution. We propose a realistic semi-automatic scheduling approach by fully utilizing available metadata associated with spatial indices and compressed data chunks. The key component of our proposed solution is a data semantic and operation semantic aware parallel data processing framework to support QDVE tasks on multi-source distributed computing resources. More specifically we propose two sets of techniques for this purpose: a set of algorithms to dynamically assemble materialized indexing data structures and a set of rules and heuristics to guide task scheduling to effectively utilize local and remote computing resources.

Sub-Task 1 Algorithms to dynamically assemble data structures from previous results. As illustrated in Fig. 1 and discussed previously, QDVE tasks on large-scale geospatial data mainly involve three data types: tree indices and raw and wavelet encoded multi-dimensional arrays. Tree indices and wavelet encoded arrays are suitable to be shipped among distributed computing resources due to their relatively small data volumes. In a way similar to TCP packages, these data structures can be split, processed and combined at distributed nodes with different types of parallel computing hardware (e.g., multicore CPUs and GPUs) before they are finally utilized at the personal workstations to interact with end users directly.

The Cache-Conscious Quadtree (CCQ-Tree) we have developed in our previous work [137] can be constructed on both multicore CPUs and GPUs. While the complete algorithms to split and combine CCQ-Tree data structures need to be developed, we next use an example to demonstrate the feasibility. Each CCQ-Tree node has a data field and a first child position field. The layout of CCQ-Tree nodes on the tree's storage array is based on a breadth-first traversal order. In Fig. 4, assuming that the two CCQ-Trees each representing a single raster layer (e.g., precipitation and temperature) at the same spatial extent need to be combined to form a multi-variable CCQ-Tree. The combination can be achieved as following through a two-pass process without requiring the respective raw multi-dimensional data arrays. During the first pass, the two trees are traversed in a synchronized manner and output the combined tree nodes to the output array. In the second pass, the positions of first child in all output nodes are set after a prefix scan process to compute the position values.

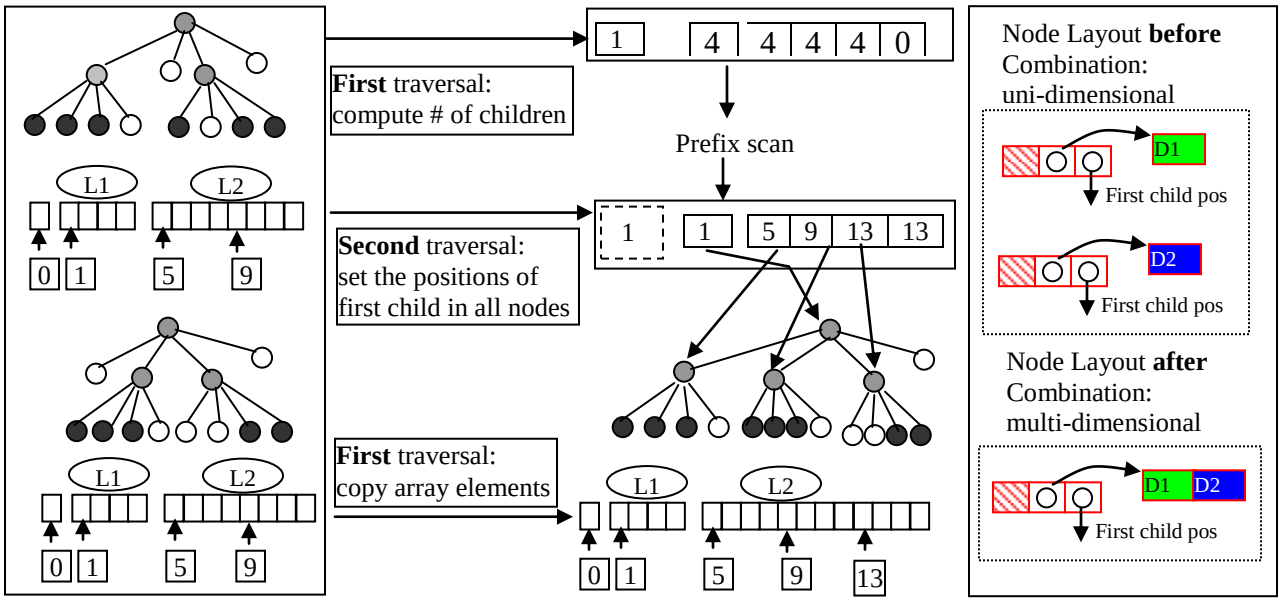

Fig. 4 Illustration of Dynamic Tree Indices Combination Using a Cache Conscious Quadtree

The example shows that it is possible to request remote computing resources to derive CCQ-Trees for the raster data that are physically collocated with processors and dynamically combine CCQ-Trees representing different raster layers to form a CCQ-Tree representing multiple raster layers for efficient query processing. High-level tree indices in our architecture (Fig. 1), although their derivation requires to process large amount of raster data, are usually small in sizes and are suitable for distributed processing without incurring significant data transfer overheads. We plan to extend the idea to develop a full set of algorithms to split and combine tree indices across both spatial domain and thematic (e.g., time, height, spectral) domains.

Sub-task 2: rules and heuristics to guide coordination of local and remote parallel computing. Given a QDVE query, based on the number and distribution of data chunks involved (usually available after performing queries on spatial indices) and the knowledge of the nature of the QDVE query, the computing intensities across the study area at the chunk level can be roughly determined. We assume the scheduling module maintains a list of data sources (including spatial indices and compressed data chunks) and computing capabilities of remote computing resources. While the rules and heuristics need to be fully developed and formalized, the following example illustrates the basic idea of our proposed

solution. Considered a QDVE application on NASA MODIS data that is similar to the draught assessment in [7] which requires find spatial locations with NDVI<0.5 AND NDWI<0.3 AND 0.5<NDDI <10. The application involves 3 out of 7 MODIS bands in a scene (red -RED, near infrared-NIR and shortwave infrared - SWIR) and three derived indices from the three bands, i.e., NDVI, NDWI and NDDI. The application also involves a few QDVE queries: two ORA queries on the RED-NIR and RED-SWIR raster pairs, respectively; one ORA queries on the NDVI-NDWI pair followed by a ROI query on the three derived indices. The data chunks satisfy the query criteria can be performed locally through set operations or synchronized tree traversals on spatial indices as the computing overhead in this filtering step is relatively light. However, to answer the queries exactly, we need to examine all the raster cells in the chunks to remove false positives. This is much more computationally intensive as both wavelet decompression and numeric calculations on the cells are involved. Assuming that a grid cluster computing has already had a copy of the MODIS data, then the refinement query step can be divided between local GPUs and remote computing. While the division will be affected by quite a few factors that need to be taken into consideration, such as the projected available grid computing resources that can be allocated to the application, network traffic delay, authentication overhead and local GPU computing power, a heuristic is to assign the chunks that are within the current view window to local GPU while assign the remaining chunks to remote computing resources based on the order of their closeness to the current view window.

We propose to distinguish two modes in parameterized QDVE queries when developing rules and heuristics, i.e., instantaneous mode and continuous mode. Different from the instantaneous mode, queries in continuous mode may change parameter values and reevaluate the queries again on a same set of raster layers. It could be more beneficial to construct some intermediate data structures to support continuous queries more efficiently. For example, while it may take extra time to construct a quadtree for the combined RED, NIR and SWIR bands from quadtrees of individual MODIS bands, querying the combined quadtree can be more efficient than querying the three quadtrees in the continuous mode. The same rule can be applied to using remote computing resources. Instead of requesting remote computing resources to send back final query results, the scheduling module may choose to ask returning wavelet encoded chunks with partial of the raw data whose value ranges are interested by the exploration tasks. In such a case, filtering non-interesting data will reduce data transfer overheads while facilitate local continuous queries efficiently.

5 Evaluation Plan

To validate the feasibility of the proposed framework and effectiveness of the proposed techniques, we propose to use three datasets with distinct characteristics and carry experiments on the four initial types of QDVE queries using a typical personal workstation. While we believe our research and development efforts on building such an end-to-end system is useful to a variety of research communities that involve handling large-scale raster geospatial data, we consider it more important to identify strong and weak points of such a system through extensive experiments and seek balances between personal computing and cloud/grid based computing in the context of QDVE applications. In addition to the data structures and algorithms we have proposed to develop, by plugging in existing open source or commercial packages (such as spatial indexing and data compression) and new hardware (such as Solid State Drive – SSD), the system can also serve as a test bed to push the limit of visual explorations of large-scale raster geospatial data in a personal computing environment.

5.1 Data Collection

We plan to use the following three real world datasets:

1) WorldClim current (1950-2000) climate data [14,15] includes 55 rasters at the global 30-arc seconds resolution as discussed in the Introduction Section. The dataset is suitable for scientists as well as the general public to explore the spatial-temporal patterns of elevation, precipitation and temperature as well as their relationships. Since the dataset is made publicly available in 2005, it has been cited 653 times according to ISI Web of Knowledge as of 12/11/2010.

(2) NASA MODIS satellite remote sensing data available at University of Maryland Global Land Cover Facility (GLCF) [142]. Seven bands are available for each of the combinations of the five geographical areas (Africa, EuraAsia, North America, South America and Oceania) over the roughly sixty 32-day periods from 2000 to 2004, i.e., 5*60*7=2100 rasters. This MODIS data has a 500 meter spatial resolution and the numbers of cells are 16700*17352, 27600*17140, 22650*15586, 11264*16032 and

18200*15400 for the five geographical areas, respectively. We plan to repeat the experiments on drought assessment and compare the performance of our personal workstation based system with that of Jaguar Cray XT4 supercomputer [7].

(3) NMQ data from University of Oklahoma and NOAA [17]. The National Mosaic & Multi-Sensor Quantitative Precipitation Estimation (NMQ) dataset covers the continental United States with a spatial resolution of 1 kilometer (roughly 5000*7000) and temporal resolution of 5 minutes (288 rasters per day). The high-temporal resolution makes it suitable for indexing and querying dynamic phenomena such as storms and other mesoscale atmospheric events. As discussed in the introduction section, we plan to use the data to explore the correlations between the deviations of NMQ and NWS observation data and the topographical factors including elevation, slope and aspect.

5.2 Experiment Settings

Our experiments will be performed on a typical commodity personal workstation, e.g., Dell T7500 with 7200-10,000 RPM hard drive, 12G - 48 G memory and an Nvidia Quadro 6000 GPU card with 448 cores and 6G graphics memory. Two of such workstations are being purchased and will soon be available at the PI's lab. A SGI Octane III two-node GPU mini-cluster that is already installed in the PI's lab will be connected with the two workstations through a 1Gbps Ethernet switch and used as on-demand computing resources to speed up visual explorations. We plan to simulate WAN applications during the initial development and test phase.

As part of the City University of New York (CUNY) system, the PI's lab is eligible to use cluster computers available at the CUNY High-Performance Computing Center (CUNY HPCC) [142]. The CCNY campus network backbone is a 1 Gbps Ethernet and is part of the CUNY network (CUNYNet) operates on an optical network with a transmission rate approximately 10 Gbps. CUNYNet, CUNY HPC and major ISPs in the New York City area connect to the collocated network hubs at the 32 Avenue Of the Americas (AOA) in Manhattan. The network topology makes it convenient for the PI's lab to connect to remote computing resources, including CUNY HPCC. We plan to use the Andy GPU cluster at the CUNY HPCC as our remote computing resources to test the proposed techniques discussed in Section 4.4. If successful, we will further explore GPU-based cloud computing resources that recently become available from Amazon [143]. In addition, we will also proactively seek support from the TeraGrid [144] and SDSC Gordon [145] teams to test our idea of using remote cluster/cloud computing as on-demand resources for QDVE applications.

5.3 Performance Metrics

While benchmarks are available for relational data (e.g., TPC [146]), we are not aware of benchmarks for raster geospatial data. Also although there are quite a few established usability assessment protocols in the information visualization community, scalability issues were not addressed until recently [32]. We plan to use the following criteria as our performance metrics: (1) End-to-end query response times for different types of QDVE queries and their combinations. Although the times will certainly vary with quite a few factors, the ranges of the response times will provide useful information on the limits of QDVE applications on personal workstations. (2) Utilizations of bandwidths (including disk I/O, CPU memory to GPU memory and GPU memory to GPU processors) and computing (including both GPU and GPU processors) for QDVE queries with different characteristics (data-intensive, computing-intensive and their combinations). (3) Usability through user survey. The focus will be whether interactive and uninterrupted visual explorations are more effective than offline modeling followed by online data display and animation.