Parallel Multivariate Spatio-Temporal Clustering of Large Ecological Datasets on Hybrid Supercomputers

Sarat Sreepathi*, Jitendra Kumar†, Richard T. Mills‡, Forrest M. Hoffman§, Vamsi Sripathi¶, William W. Hargrove‖

*Computer Science and Mathematics Division, Oak Ridge National Laboratory,
Oak Ridge, TN, USA Email: sarat@ornl.gov
†Environmental Sciences Division, Oak Ridge National Laboratory,
Oak Ridge, TN, USA Email: jkumar@climatemodeling.org
‡Mathematics and Computer Science Division, Argonne National Laboratory,
Lemont, IL, USA Email: rtmills@anl.gov
§Computational Science and Engineering Division, Oak Ridge National Laboratory,
Oak Ridge, TN, USA Email: forrest@climatemodeling.org
¶Intel Corporation, Hillsboro, OR, USA Email: vamsi.sripathi@intel.com
‖Eastern Forest Environmental Threat Assessment Center, USDA Forest Service,
Asheville, NC USA Email: hnw@geobabble.org

***Abstract*—A proliferation of data from vast networks of remote sensing platforms (satellites, unmanned aircraft systems (UAS), airborne etc.), observational facilities (meteorological, eddy covariance etc.), state-of-the-art sensors, and simulation models offer unprecedented opportunities for scientific discovery. Unsupervised classification is a widely applied data mining approach to derive insights from such data. However, classification of very large data sets is a complex computational problem that requires efficient numerical algorithms and implementations on high performance computing (HPC) platforms. Additionally, increasing power, space, cooling and efficiency requirements has led to the deployment of hybrid supercomputing platforms with complex architectures and memory hierarchies like the Titan system at Oak Ridge National Laboratory. The advent of such accelerated computing architectures offers new challenges and opportunities for big data analytics in general and specifically, large scale cluster analysis in our case. Although there is an existing body of work on parallel cluster analysis, those approaches do not fully meet the needs imposed by the nature and size of our large data sets. Moreover, they had scaling limitations and were mostly limited to traditional distributed memory computing platforms. We present a parallel Multivariate Spatio-Temporal Clustering (MSTC) technique based on k-means cluster analysis that can target hybrid supercomputers like Titan. We developed a hybrid MPI, CUDA and OpenACC implementation that can utilize both CPU and GPU resources on computational nodes. We describe performance results on Titan that demonstrate the scalability and efficacy of our approach in processing large ecological data sets.**

I. Introduction

Earth science data captures numerous nonlinear and complex interactions among high dimensional set of variables representing wide range of ecosystems processes. Classification is one of the most widely used statistical methods in ecology for development of ecoregions [1], classification of climate zones [2], mapping of vegetation using remote sensing [3], characterization of vegetation structure [4], and species distribution modeling [5]. Quantitative methods for classification, including multi-variate cluster analysis [6] and random forests [7], are increasingly used to statistically explore and exploit multi-variate relationships in such rich data sets.

Earth science data has seen a rapid increase in both complexity and volume over the recent decade. These growing volumes of data range from field and laboratory based studies to environmental sensor network to ground, air and space based remote sensing platforms. These data sets offer new opportunities for scientific discovery. However, the volume and complexity of the data has also rendered traditional means of integration and analysis ineffective, necessitating the application of new analysis methods and the development of highly scalable software tools for synthesis, comparison, and visualization [8].

This manuscript has been authored by UT-Battelle, LLC under Contract No. DE-AC05-00OR22725 with the U.S. Department of Energy. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan(http://energy.gov/downloads/doe-public-access-plan).

Large and complex Earth science data often cannot be synthesized and analyzed using traditional methods or on individual workstations. Data mining, machine learning, and high performance visualization approaches are increasingly filling this void and can often be deployed only on parallel clusters or supercomputers. However, supercomputer architectures designed for compute-intensive simulations, usually containing large numbers of cores with high speed interconnects between nodes, are not typically optimal for large scale analytics. Instead, such applications demand large and fast on-node memory, high bandwidth input/output (I/O), and fast access to large local disk volumes. Most domain scientists are ill-equipped to develop analytics codes for these architectures, while system vendors have largely focused on compute-intensive applications, and must acquire representative analytics benchmarks and scientific expertise to design systems for geospatial big data analytics.

A. Related Work

A number of studies in past have developed parallel cluster analysis implementations targeting range of data sets and computing platforms. [9] designed an implementation for clustering algorithms for Beowulf-style parallel cluster built from surplus computer. [10] designed implementation of cluster analysis for mid-range distributed memory cluster using a master-slave paradigm. A number of other works by [11]–[14] have developed approaches for efficient implementations of parallel cluster algorithms to analyze large data sets, however most of them have focused on traditional CPU based distributed memory supercomputers. New generations of supercomputers, like Titan at Oak Ridge National Laboratory and its planned successor, Summit are based on GPU-based hybrid architectures. There exist several studies [15]–[17] that have looked into accelerating k-means on the GPUs. However, the dimensionality and size of our target datasets are relatively larger in comparison and warrant specialized preprocessing and normalization. Moreover, we are striving for a faster time to solution by utilizing all the available computational resources, CPUs and GPUs on a node in tandem. Hence, the focus of this study was to improve and adapt our k-means clustering algorithm on hybrid architectures for large earth science data to provide a scalable parallel cluster analysis tool for next generation supercomputing architectures in general and U.S Department of Energy's leadership class supercomputers in particular.

II. Data sets and Experiment Setup

A. Data sets

Tools and methods developed in this study were applied and tested for two earth science applications (Table I).

1) Vegetation structure of Great Smoky Mountains National Park (`GSMNP`): Understanding of vegetation structure of forest ecosystem is key for forest health management and maintaining suitable habitats for bird and animal species. Airborne multiple return Light Detection and Ranging (LiDAR) data for GSMNP [18] provides high resolution view of the three-dimensional structure of the forest ecosystem. Raw LiDAR point clouds were processed to develop vertical canopy structure of the vegetation at $30\ m\ \times\ 30\ m$ spatial resolution horizontal grid and $1\ m$ resolution [4]. A $1\ m$ vertical resolution was used to identify vegetation height from the ground surface to a maximum height of $75\ m$. The number of LiDAR points in each vertical $1\ m$ bin (at each $30\ m\ \times\ 30\ m$ cell in the horizontal grid) was identified to construct a vertical density profile for each map cell. Classification of LiDAR derived vegetation structure is desired to understand the spatial pattern and distribution of vegetation structure across the GSMNP.

TABLE I
Description of data sets used in the current study

Description	Dimensions	Size
GSMNP LiDAR	3,186,679 × 74	900 MB
CMIP3 Climate States	123,471,198 × 17	7.9 GB

2) Global Climate Regimes (`GCR`): Classification of climate regimes has long been used to understand the global patterns of climate, vegetation and terrestrial ecology. We want to understand and analyze the climate regimes in contemporary period and how they may change and shift in future under various predicted climate change scenarios. We selected a range of bioclimatic, edaphic and topographic variables globally at 2 $arcsecond$ ($\sim\ 4\ km$) resolution to define the climate regimes. Bioclimatic data for the contemporary period were derived from BioClim data sets by [19]. To represent future climate, two climate models from the Intergovernmental Panel on Climate Change Third Assessment Report (CMIP3) – Parallel Climate Model (PCM) developed by National Center for Atmospheric Research and HadCM3 model developed by Hadley Center, were used. Model data for two different emissions scenarios, B1 (lower emissions) and A1FI (high emissions) were used and bioclimatic variables were derived (Table II) for two select future periods (2050, 2100) [20], [21].

B. Preprocessing

Large ecological data sets often suffer from data noise, errors and missing values. All the data sets used in the study were carefully checked, corrected and gap filled. Heterogeneity among high dimensional data sets is typical of ecological and earth science data sets. `GSMNP` data set was derived from LiDAR point clouds and was homogeneous across all the dimensions. However, the 17 dimensions of the `GCR` data each represent a different physical quantity with different scales and units. We standardized the data set along each dimension to have a mean of zero and standard deviation of one, allowing every dimension to be equally and fairly represented in the clustering algorithm.

III. Methodology

In this section, we describe our baseline k-means algorithm for clustering and an algorithmic scheme using triangle inequality for reducing the number of distance calculations.

TABLE II
VARIABLES USED FOR DELINEATION OF GLOBAL CLIMATE REGIMES.

Variable Description	Units
Bioclimatic Variables	
Precipitation during the hottest quarter	mm
Precipitation during the coldest quarter	mm
Precipitation during the driest quarter	mm
Precipitation during the wettest quarter	mm
Ratio of precipitation to potential evapotranspiration	–
Temperature during the coldest quarter	°C
Temperature during the hottest quarter	°C
Day/night diurnal temperature difference	°C
Sum of monthly T_{avg} where $T_{\text{avg}} \geq 5$°C	°C
Integer number of consecutive months where $T_{\text{avg}} \geq 5$°C	–
Edaphic Variables	
Available water holding capacity of soil	mm
Bulk density of soil	g/cm^3
Carbon content of soil	g/cm^2
Nitrogen content of soil	g/cm^2
Topographic Variables	
Compound topographic index (relative wetness)	–
Solar interception	(kW/m^2)
Elevation	m

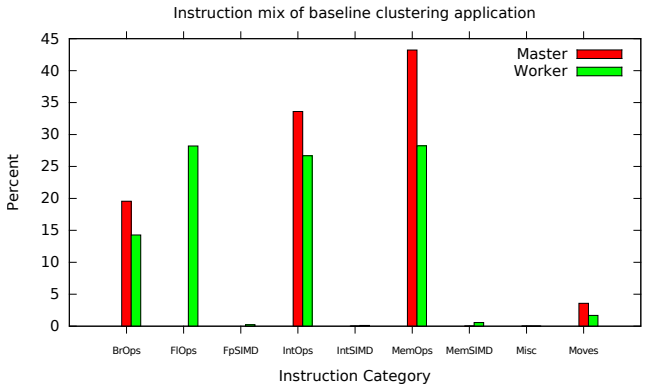

Fig. 1. The instruction mix for the baseline application while running on 16 processors using the GSMNP data set. The red bar corresponds to the master process that primarily handles communication, explaining the lack of any floating point operations. The green bar represents worker processes that exclusively handle the computation, as reflected in floating point operations.

A. Baseline k-means algorithm

The k-means is iterative algorithm to group a data set (X_1, X_2, …, X_n) with n records into desired k clusters. k-means algorithm groups the data into desired number of groups while equalizing the multi-dimensional variance across clusters. The algorithm starts with a set of initial "seed" centroids (C_1, C_2, …, C_k), and calculated the Euclidean distance of each data record (X_i, $1 \leq i \leq n$) to every "seed" centroid (C_j, $1 \leq j \leq k$), Data record is classified to the cluster containing the closest existing centroid. After all data records are classified, a new centroid is calculated as the mean vector of all dimensions of each data record classified to that cluster. As this cluster assignment and re-calculation of centroid is iteratively repeated, the centroids move through the data space to identify stable, and optimal values such no more than a small proportion (we use $< 0.05\%$) of data records change their cluster assignments between iterations.

B. Accelerated k-means using triangle inequality

We also implemented a triangle inequality [22], [23] based acceleration scheme that reduces the number of Euclidean distance calculations. Triangular inequality states: for any three points x, y, and z, $d(x,z) \leq d(x,y) + d(y,z)$. The algorithm eliminates unnecessary point-to-centroid distance calculations and comparisons based on the previous cluster assignment and the new inter-centroid distances.

If the distance ($d(C_{last}, C_{new})$) between the last centroid (C_{last}) and new candidate centroid (C_{new}) and greater than or equal to the distance ($d(X_i, C_{last})$) between a data point (X_i) and the last centroid (C_{last}), then calculation of distance ($d(X_i, C_{new})$) between the data point (X_i) and the new candidate centroid (C_{new}) can be avoided. Triangle inequality states that $d(C_{last}, C_{new}) \leq d(X_i, C_{last} + d(X_i, C_{new})$. If $d(C_{last}, C_{new}) \geq 2d(X_i, C_{last}$, we can conclude without calculating $d(X_i, C_{new})$, that $d(X_i, C_{new}) \geq d(X_i, C_{last})$. Thus, for the data point, X_i, the new centroid candidate (C_{new}) can be eliminated without computing the distance $d(X_i, C_{new})$.

The Euclidean distance computations can be further reduced by sorting the inter-centroid distance ($d(C_{last}, C_{new})$). The new candidate centroids (C_{new} are evaluated as per sorted distance order, and once the critical distance ($2d(X_i, C_{last})$ is surpassed all subsequent candidate centroids can be safely discarded without any distance calculations.

IV. BASELINE PERFORMANCE CHARACTERIZATION

We collected performance data with our baseline clustering implementation using the LiDAR data set for the Great Smoky Mountains National Park (GSMNP).

- We utilized the Oxbow toolkit and Performance Analytics Data Store (PADS) [24] infrastructure for this application characterization.
- This kind of data is invaluable to identify potential opportunities for improvement and aid in adaptation to emerging architectural features.

A. Computational Profiling

The computational profile of application execution is described by the mix of executed micro-operations. Figure 1 shows the instruction mix of our clustering application.

- Obtained by decoding the x86 assembler instructions and grouping them into coarser categories like memory, control, floating point and integer arithmetic.
- Obtained using a tool based on Intel's PIN [25], a dynamic binary instrumentation tool.
- The data is useful to ascertain if there is potential for improved performance. For instance, we identified an opportunity for improved performance by better utilization of floating point operations including single-instruction-multiple-data (SIMD) operations which led to the development of the distance calculation using BLAS formulation as described in Section V-A.

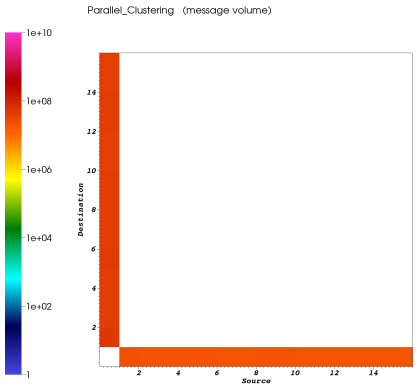

Fig. 2. Communication volume for baseline clustering algorithm using 16 MPI processes. The axes show the ranks of the sender and receiver process respectively.

B. Communication Behavior

We used augmented version of the communication profiling tool (mpiP) [26] to capture the volume of data transferred between MPI ranks and visualized the results to understand the communication topology (Figure 2). It is evident that we are using a master-worker protocol because all communication is point-to-point between the first process and rest of the processes.

C. Memory Behavior

We instrumented the kernel of our application using PAPI hardware counters for obtaining detailed memory performance data. The kernel achieves a read bandwidth of 122 MB/s and a write bandwidth of 58.9 MB/s. These results are for the baseline application with no in-memory data rearrangement to optimize memory performance.

V. Optimizations

This section elaborates on the recent additions for improving performance by using a more efficient problem formulation for computing distances between observations and centroids, as well as threading support.

A. Distance Calculation: Using BLAS

Our clustering code has, for years, calculated observation–centroid differences one at a time (necessary to employ the "acceleration" technique previously described). Recently, one of the authors realized that it is possible to achieve much greater computational intensity in the observation–centroid distance calculations by expressing the calculations in matrix form. This enables the use of level-2 and level-3 BLAS routines, for which highly cache-optimized implementations that have also been tuned to make good use of SIMD instructions, etc., are available, and also facilitates the use of compute accelerators like GPGPUs (general purpose graphical processing units).

Internally, our clustering code stores observation vectors as rows in a matrix, so we adopt that convention here. Let **obs** be the observation matrix that contains n observations of m dimensions, and **cent** be the centroid matrix that contains the k desired centroids and their coordinates in m dimensions.

We wish to compute the $n \times k$ matrix of squared Euclidean distance, **dist**, for which the i, jth entry

$$\mathbf{dist}_{i,j} = \|\mathbf{obs}_{i,*} - \mathbf{cent}_{i,*}\|^2 \quad (1)$$

contains the squared Euclidean distance between observation i and centroid j. The key insight to reformulating the distance calculation in matrix form is that, via binomial expansion,

$$\mathbf{dist}_{i,j} = \|\mathbf{obs}_{i,*}\|^2 + \|\mathbf{cent}_{i,*}\|^2 - 2 \cdot \mathbf{obs}_{i,*} \cdot \mathbf{cent}_{j,*} \quad (2)$$

and, therefore, we can express

$$\mathbf{dist} = \overline{\mathbf{obs}} \cdot \mathbf{1}^T + \mathbf{1} \cdot \overline{\mathbf{cent}}^T - 2 \cdot \mathbf{obs} \cdot \mathbf{cent}^T \quad (3)$$

where $\overline{\mathbf{obs}}$ and $\overline{\mathbf{cent}}$ are vectors of the sums of all squares of the rows of **obs** and **cent**, respectively, and $\mathbf{1}$ is a vector of all 1s.

Formulated as above, we utilize BLAS routines as follows to calculate the matrix of squared Euclidean distances:

1) Calculate $-2 \cdot \mathbf{obs} \cdot \mathbf{cent}^T$ via *xGEMM*, the level-3 general matrix-matrix multiplication subroutine that computes

   $$C := alpha * op(A) * op(B) + beta * C$$

   Where *alpha* and *beta* are scalars, A, B, and C are matrices and *op* optionally performs matrix transpose or conjugate transpose.
2) After the xGEMM operation, use the level-2 BLAS routine *xGER*, to add $\overline{\mathbf{obs}} \cdot \mathbf{1}^T$ and $\mathbf{1} \cdot \overline{\mathbf{cent}}^T$ via a rank-one update, of the form

   $$A := alpha * x * y' + A$$

   Here, *alpha* is a scalar, x, y are element vectors and A is the input matrix.

Casting the distance calculation into the form of level 2 and (especially) level 3 BLAS operations facilitates the use of highly computationally efficient implementations. Because we use standardized BLAS interfaces, we are able to use vendor-optimized BLAS libraries—such as Cray's LibSci, Intel's MKL, and IBM's ESSL—on their respective systems.

Our experiments using the above matrix formulation for the distance calculations show that, as expected, it is dramatically faster than the straightforward loop over vector distance calculations when many distance comparisons must be made. We give details in Section VIII. For architectures that employ a high level of fine-grained parallelism with wide SIMD lanes, increasing the computation intensity has an especially high payoff in terms of improved performance. In a future paper, we will discuss the performance of this implementation on one such architecture, the second-generation Intel Xeon Phi ("Knights Landing") processor, where the matrix formulation is especially advantageous and can beat the triangle inequality-based "acceleration" technique in several situations, despite performing many more distance calculations within a k-means iteration.

B. Application Phases

During the initial phase of the application, a large number of pairwise distances between observations and centroids need to be computed resulting in a relatively higher number of changes in cluster assignments for the observations. This phase is particularly suitable for distance matrix computation using the BLAS formulation. Once the clusters stabilize, there are fewer changes and the triangle inequality based acceleration technique obviates the need for computing the full distance matrix using the BLAS formulation. We have empirically determined the transition points between these two application phases for specific data sets and switch from the BLAS formulation to the triangle inequality method. We intend to add the capability to identify this phase transition during runtime in the future.

C. Vectorization and OpenMP

We have added SIMD compiler directives for vectorization where applicable. Although the clustering code already employed full distributed-memory parallelism via MPI, we added threading support and used dynamic thread scheduling for the triangle inequality acceleration component, which enables better use of all available hardware threads on architectures such as the second-generation Intel Xeon Phi processor. Due to the requisite updates and branching involved, we incorporated a critical region to ensure correctness.

VI. Targeting GPUs

This section details our application porting work to the GPUs using cuBLAS and OpenACC kernels.

A. cuBLAS

We utilized NVIDIA's cuBLAS [27] library on the GPUs. Our application uses row-major ordering for the major data structures as it is written in the C programming language. Hence, we had to modify our arguments to the cuBLAS subroutines as it assumes the Fortran column-major ordering for matrices.

We developed a standalone kernel and conducted a detailed performance analysis after incorporating the cuBLAS calls. Table III shows the performance profile for the GPU kernel using the GCR dataset.

TABLE III
Performance profile of our cuBLAS GPU kernel

Time(%)	Avg. Time	Calls	Name
97.20	9.89 s	1	[CUDA memcpy DtoH]
1.41	71.8 ms	2	void ger_kernel
0.77	78.18 ms	1	sgemm_sm_heavy_nn_ldg
0.37	6.22 ms	6	[CUDA memcpy HtoD]
0.26	26.43 ms	1	sgemm_sm35_ldg_nn_64x16x128x8x32
0.00	33.69 us	1	sgemm_sm35_ldg_nn_128x16x64x16x16

We identified the copying back of the pairwise distance matrix from the GPU back to the host CPU as the major performance bottleneck. We decided to perform the requisite post-processing of the distance matrix on the GPU itself to avoid copying the matrix back to host. This effort is described in detail in the next section.

B. OpenACC additions

We implemented a couple of OpenACC kernels to post-process **dist**, the pairwise squared distance matrix of observations and centroids on the GPU itself. This is required to update the cluster assignments for the observations in addition to bookkeeping tasks to keep track of farthest observation in each cluster. This process entails operations such as finding the minimum value and index for each row and maximum value and index for each column.

C. Verification

We performed unit testing at every step to ensure the accuracy of the new kernels. It is infeasible to achieve bit-for-bit reproducibility due to variations in floating point arithmetic in BLAS libraries, etc. However, we have verified the final cluster assignments in a quantitative manner (numerical comparison) and qualitatively by generating maps of the final clustering results.

VII. Computational Platform

We conducted our experiments on Titan [28], a Cray supercomputer installed at Oak Ridge National Laboratory (ORNL). Titan is a hybrid-architecture Cray XK7 system with a theoretical peak performance exceeding 27 petaflops. It comprises of 18,688 compute nodes, wherein each node contains 16-core AMD Opteron CPUs and NVIDIA Kepler K20X GPUs for a total of 299,008 CPU cores and 18,688 GPUs. Each node has 32 GB memory that amounts to 2 GB/CPU core. Additionally, there is 6 GB of memory available on the GPU. It has a total system memory of 710 terabytes, and utilizes Cray's high-performance Gemini interconnect. Titan has a $25 \times 16 \times 24$ 3D torus network where 2 compute nodes share a network interface. As of November 2016, it is the third fastest supercomputer in the world according to the TOP500 list [29].

The software environment for the reported experiments is as follows: Cray PGI programming environment (version 5.2.82) which uses PGI 16.10.0 compilers and Cray's MPICH implementation (version 7.5.2). We utilized Intel's MKL (Math Kernel Library) for BLAS matrix operations on the host CPU. We used cuBLAS and CUDA toolkit (version 7.5.18-1.0502.10743.2.1) for GPU programming.

VIII. Computational Performance

We performed several experiments on Titan using the large GCR data set and different problem configurations. The performance gains from our optimization efforts are demonstrated in figure 3. In this scenario, we are comparing the performance of the baseline application with the optimized version while using the large GCR data set to find 8,000 clusters till a specified convergence target is reached. We used a target of 5% or fewer changes in cluster memberships between iterations as the termination criteria for the performance experiments. We use a better threshold (0.5%) for higher fidelity scientific experiments. Note that the optimized version yields a speedup of $2.7\times$ over the baseline version. The application spends

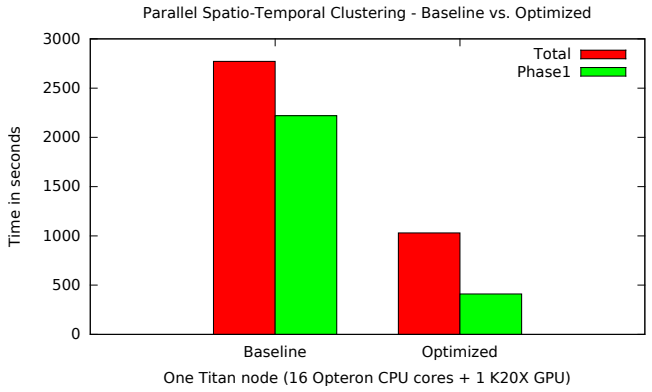

Fig. 3. Parallel Spatio-Temporal Clustering : Performance comparison of the Baseline application with the Optimized version for finding 8,000 clusters using the GCR data set on one node of Titan. A speedup of 2.7× is observed with the Optimized version.

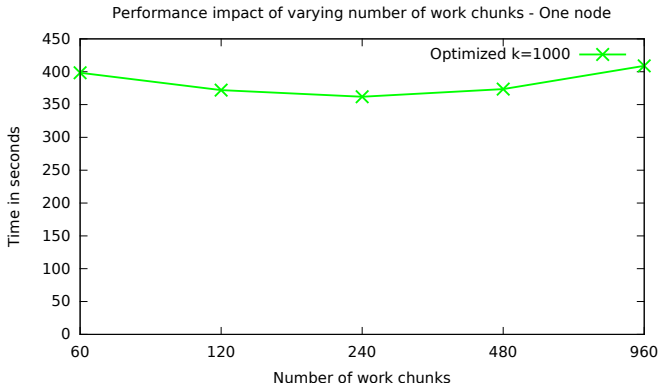

Fig. 5. Parallel Spatio-Temporal Clustering: Experimenting with number of work chunks used for load balancing among non-homogeneous workers (CPU and GPU) using the GCR data set for finding 1,000 clusters. A chunk size of 240 results in comparatively better performance for this case.

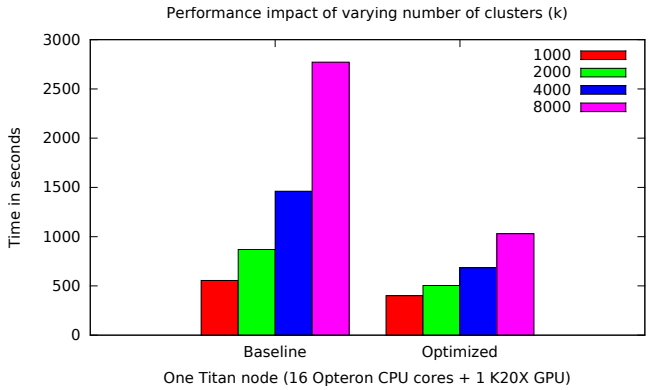

Fig. 4. Parallel Spatio-Temporal Clustering : Performance impact of the parameter k (number of clusters) using the GCR data set on one node of Titan. Please note that performance gains with optimized implementation are more conspicuous with larger clusters due to increased computation.

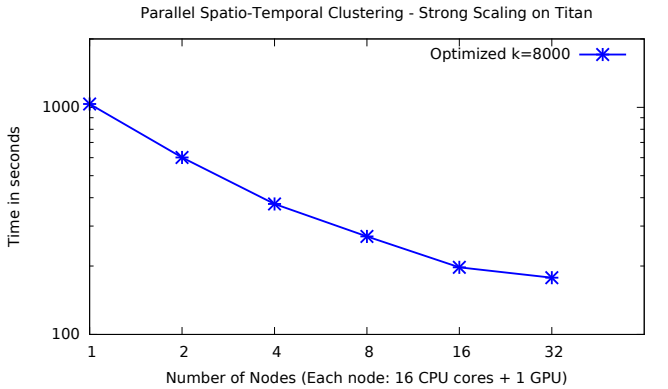

Fig. 6. Parallel Spatio-Temporal Clustering : Strong scaling performance for finding 8,000 clusters using the GCR data set on Titan. The scaling is limited at higher node counts due to insufficient computational workload per process.

a majority of time in the first phase (Phase1), and the substantial improvement stems from accelerating Phase1 using GPUs.

A. Impact of k

The desired number of clusters (k) has significant influence on application execution time. We conducted several experiments to quantify this impact as shown in figure 4. The performance benefits of the optimized version become more prominent as k increases due to the increased computational intensity of the application.

B. Dynamic Load Balancing

We have a centralized master process that allocates work dynamically to both CPU and GPU workers. At every iteration, the master process distributes initial chunks to available workers and assigns next chunk upon completion. We can vary the number of chunks of work, or *aliquots*, per iteration using a parameter naliquot for effective load balancing between non-homogeneous workers . The impact of this parameter on performance is shown in figure 5 using a single node of Titan for the problem of finding 1,000 clusters for the GCR data set. Although there is low degree of variability, we can observe that a chunk size of 240 seems optimal for this particular problem configuration.

C. Scaling

The strong scaling performance of our parallel clustering implementation is shown in figure 6 for the 8,000 cluster scenario. For this problem configuration, the application scales well to sixteen nodes for a total of 256 CPU cores + 16 GPUs. It must be noted that there is insufficient computation for the 8,000 cluster problem to amortize the communication and data distribution overheads at larger node counts.

D. Limitations and Future Work

Our current approach uses a centralized master process to coordinate and keep track of worker processes. If used with a sufficiently small chunk (*aliquot*) size, this provides dynamic load balancing, which is especially useful when employing the triangle inequality-based acceleration technique, as the number

of required distance comparisons will vary between chunks. The centralized master-worker paradigm has inherent scalability limits, however, and introduces a large amount of overhead when many processes are used; furthermore, for certain large data sets or problem configurations with higher number of desired clusters, the memory requirements for storing the cluster assignment table and intermediate data structures will exceed the available memory on a node, which limits what we can analyze on Titan. For these reasons, we plan to add support for a decentralized approach (which we have explored using a different version of the clustering implementation [11]). Furthermore, we are interested in using non-volatile memory (NVM)—which promises very large amounts of byte-addressible memory—to store the cluster assignment table and other applicable data structures.

One of our key optimizations has been the use of level-2 and level-3 BLAS routines using the matrix formulation of the distance calculations. We currently combine this with the triangle inequality-based acceleration in a crude manner by simply specifying an iteration count at which to switch from using the former approach to the latter. Developing a heuristic to automatically select when this transition should occur is one possible improvement. It may be feasible to do something more sophisticated and combine the two approaches, performing all distance calculations in initial iterations via the matrix approach, and then, as cluster memberships stabilize, using the matrix formulation for calculations using only a subset of the centroids.

IX. Applications

A. Vegetation structure of Great Smoky Mountains National Park

LiDAR based vertical density profiles of vegetation in GSMNP were classified among 30 clusters to identify distinct vegetation structure type within the park. Choice of 30 clusters in our study was based on [4]. Figure 7 show the 30 representative vertical structures (cluster centroids) identified by the cluster algorithm. For example, cluster 1 represent tall forests with mean height of $30-40\ m$ but with low understory vegetation, while cluster 2 represent forests with slightly lower mean height of $25-30\ m$, but with a dense understory vegetation under $10\ m$. Clusters 13, 14, 20 represent low height grasslands and heath balds that are small in area but distinct landscape type within the GSMNP. While most of the past LiDAR based studies of forest ecosystems focus primarily on the maximum canopy height derived from the point clouds, our clustering based analysis identifies and highlights the immense diversity in vertical structure of the vegetation (Figure 7) of different height, density and stature across the park.

Figure 8 show the spatial distribution of the 30 vegetation clusters across the national park. Structural complexity of the vegetation in GSMNP across the gradients of topography, precipitation and moisture availability and climate expressed through diversity in vegetation species composition is visualized in the Figure 8. High elevation regions of the park are

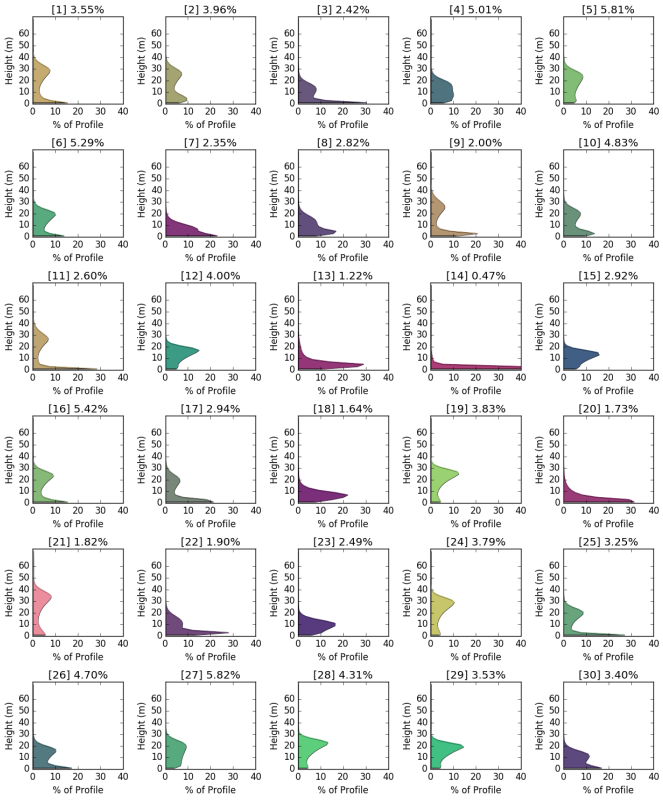

Fig. 7. Representative vegetation structure profiles identified by k-means cluster algorithm (k=30) across GSMNP. Each vegetation profile show a normalized density distribution of the vegetation bioss in the vertical canopy. Also shown for each cluster is the fraction of total land area within the park which it occupies.

dominated by the short height vegetation canopies with dense understory. Vegetation at these high elevations are subjected to harsher climate conditions and are thus dominated by relatively shorter tree canopies with dense understory shrubs like Rhododendron and Mountain Laurels. Tall canopy vegetation are prominent in mid to low elevation mountain coves, especially on northern aspect mesic slopes that provides high moisture and radiation environment to support tall vegetation species in the park. Analysis of the entire vertical canopy, unlike the maximum height in most previous studies, reveals spatial patterns of vegetation structure that are influenced by microclimatic conditions leading to a great range of diversity not just across different vegetation types and species but also within same species and forest types. These patterns provides insights in the range of climate conditions a given species grow in and adapt to and is indicative of vegetation health and diversity.

B. Global climate regimes

Tremendous amount of heterogeneity in terms of climate, vegetation, soil properties and nutrients and topography exist in the terrestrial land ecosystem across the globe. At the same time similarities in environmental conditions exist at regional scales and at times across regions that may be geographically

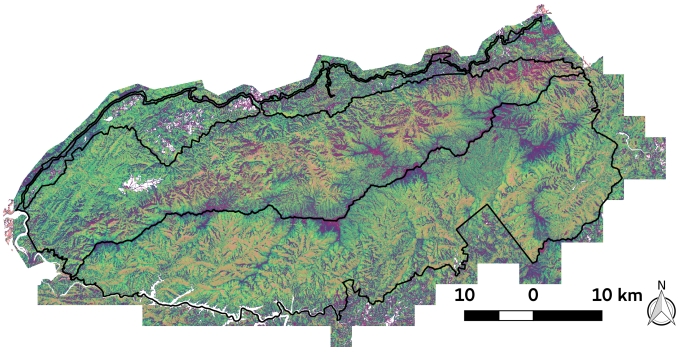

Fig. 8. Spatial distribution of 30 vegetation structure classes/clusters (Figure 7) across the Great Smoky Mountains National. Boundaries of the GSMNP are shown by black lines on the map. The black line across the middle of the park following the ridge line of mountains is the state line with Tennessee in north and North Carolina to south of it. Color scheme is the map corresponds to the color scheme for cluster in Figure 7.

disconnected and distant. Goal of global climate regimes is to characterize the environmental conditions descried by multi-dimensional data sets (Table II) in a set of cohesive data defined regions and help quantify the large scale patterns of climate and environment.

Level of divisions (k) in k-means clustering provides resolution in multi-dimensional data space, that can be tuned depending on the specific resolution. Figure 9 show map of 1000 climate regimes identified by k-means clustering using multi-dimensional data sets (Table II). Clustering is able to identify biomes all across the globe, like, Appalachian mountains in eastern United States, agricultural regions in United States mid-west, Alaskan boreal forests, dry and wet tropical forests in Amazon etc. However, while clustering can characterize the heterogeneities in the complex data sets well, visualization of the results for analysis purposes pose a unique challenge. Colored using random colors, Figure 9 is difficult to interpret. Generating 1000 distinctly identifiable colors for visualization is a difficult problem, bound by the limitations of human eye to perceive colors.

We quantitatively generated color schemes (*similarity colors*) for the map that embeds the environmental conditions in the color used for the maps, making their interpretation easy and intuitive. We performed a Principal Component Analysis (PCA) on the final centroids identified for the 1000 clusters by the k-means algorithm. The first three principal components (PCs) explain 62% of the total variance in the data. First principal component (PC1) represents 30% of the variance and was dominated by precipitation related variables and evapotranspiration. Second principal component (PC2) was dominated by temperature variables and length of growing season and explained 20% of total variance. Third principal component (PC3) primarily represented solar radiation, topography and soil nutrient variables and explained 12% of the total variance. Values of first three principal components were used to generate RGB color schemes for the map. PC1 was assigned to Green channel, PC2 to Blue and PC3 was assigned to the Red channel to generate the *similarity colors*. Figure 10 shows the same map as Figure 9 but using *similarity colors*. While Figure 9 highlights the boundaries between climate regimes well, Figure 10 uses a continuous color scheme that highlights the dominant environmental conditions (based on PCs) that characterizes the regime. Northern hemisphere temperate and high latitudes are dominated by temperature variables. Effect of precipitation and soils are visible in eastern United States, and topographic complexities of Sierra Nevada and Rocky Mountains in western United States are depicted by complexity of colors on the map. Precipitation and soil conditions are increasingly dominant in tropical region in both hemispheres and latitudinal similarities across the continents in tropical region are prominent.

Analysis of future climate regimes using two climate models and two different climate scenarios show key shifts expected in the large scale climate regimes globally under climate change scenarios. Due to space limitations, we present results only for HadCM3 climate model under A1FI scenario in 2100 (Figure 11). A northward shift in regimes can be observed, especially in northern hemisphere temperate zones under in future warming climate. The changes in the climate regimes are especially prominent in tropical regions due to expected shift in precipitation patterns and warmer climate.

X. Conclusion

In this paper, we presented a parallel multivariate spatio-temporal clustering algorithm and its application to processing big data sets in ecology. Through a detailed performance characterization of our application, we identified the need to increase the computational intensity to achieve better performance on advanced architectures. Towards that end, we implemented a high performance BLAS formulation to accelerate Euclidean distance calculations that formed the dominant component of our baseline application. We have made substantial efforts to improve the performance of the baseline algorithm by utilizing all the computational resources available on hybrid supercomputers. Using a combination of MPI, CUDA and OpenACC, we demonstrated up to 2.7X speedup in certain problem configurations with the optimized implementation on the Titan supercomputer at Oak Ridge National Laboratory. We applied our technique and demonstrated efficacy in addressing two of Earth science problems, namely (a) Great Smoky Mountains National Park: identification of vegetation structure and (b) Globlal Climate Regimes: understanding the global patterns of climate, vegetation and terrestrial ecology.

Our future plans include (a) design of a decentralized version to overcome scalability limitations with large process counts and memory limitations with large data sets, (b) experimenting with non-volatile memory technologies for storing cluster assignments and intermediate data structures, (c) better integration of matrix and triangle inequality-based formulations of the distance calculations, and (d) techniques for effective utilization of fat hybrid nodes like those present in the next generation supercomputer, Summit, which will have multiple GPUs per node.

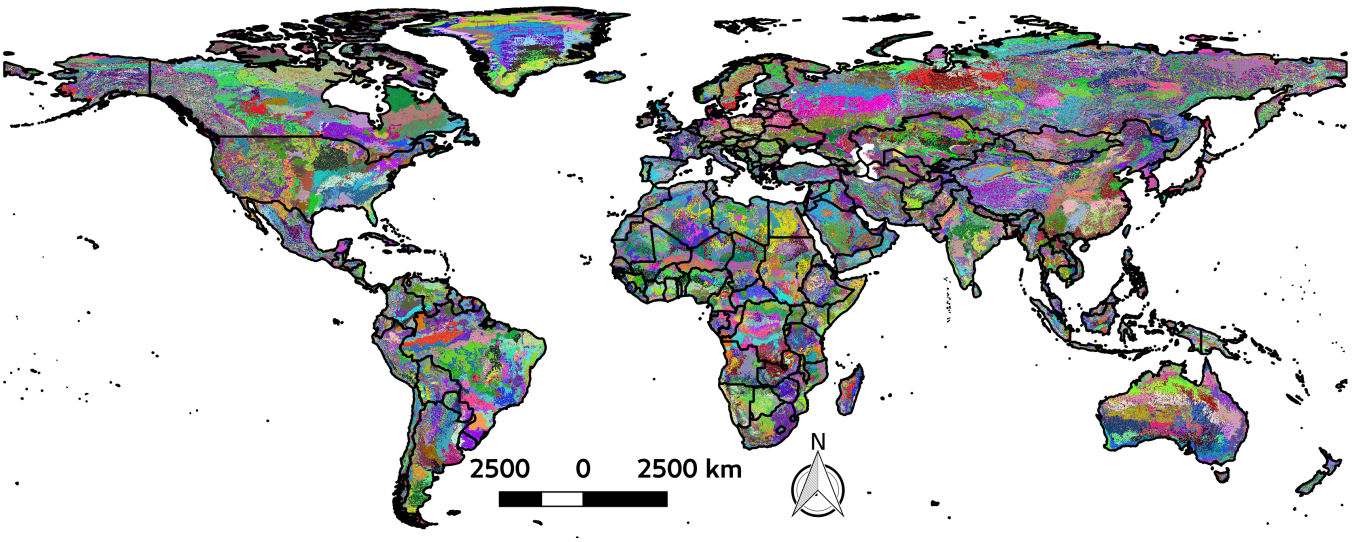

Fig. 9. 1000 Global climate regimes generated by the k-means clustering algorithm for contemporary time period. Randomly generated colors were assigned to each cluster to highlight the extent and boundaries among the climate regimes.

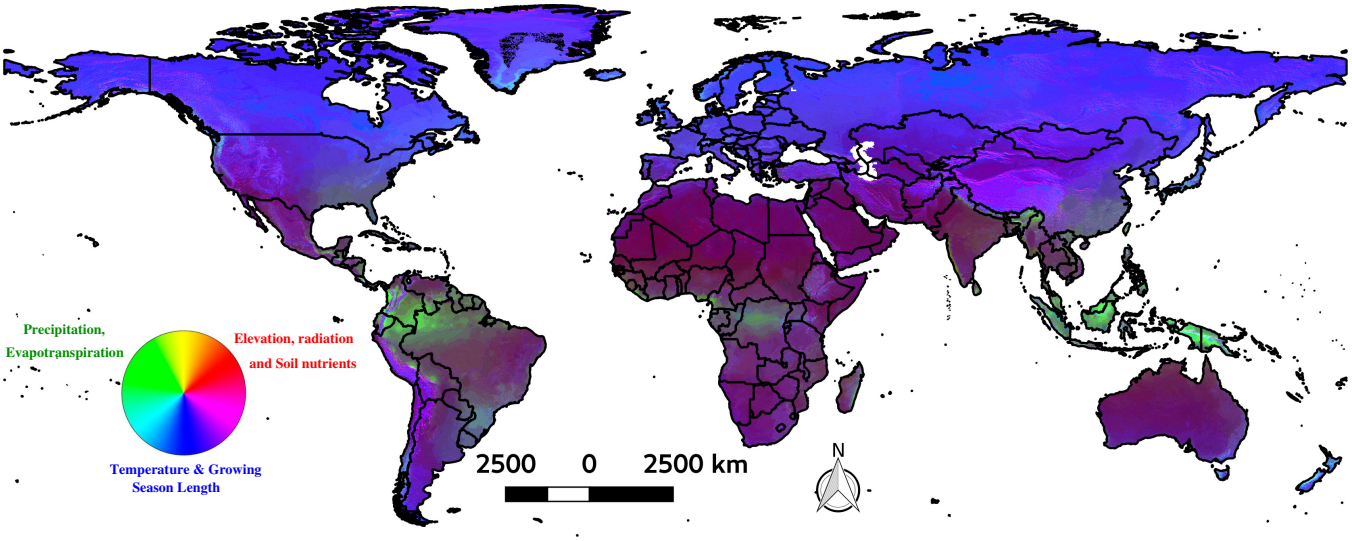

Fig. 10. 1000 Global climate regimes generated by the k-means clustering algorithm (same as Figure 9) for contemporary time period. *Similarity color* scheme was used where Red color channel highlights effect of topography and soil properties, Green channel highlight precipitation variables and evapotranspiration, and Blue channel demonstrate the effect of temperature variables and growing season length.

ACKNOWLEDGMENT

This research was partially supported by the U.S. Department of Agriculture, U.S. Forest Service, Eastern Forest Environmental Threat Assessment Center. Partial support for this work was provided through the Scientific Discovery through Advanced Computing (SciDAC) program funded by the U.S. Department of Energy Office of Advanced Scientific Computing Research (ASCR). Awards of computer time was provided by the Innovative and Novel Computational Impact on Theory and Experiment (INCITE) program. This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725. This manuscript has been authored by UT-Battelle, LLC under Contract No. DE-AC05-00OR22725 with the U.S. Department of Energy. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan(http://energy.gov/downloads/doe-public-access-plan).

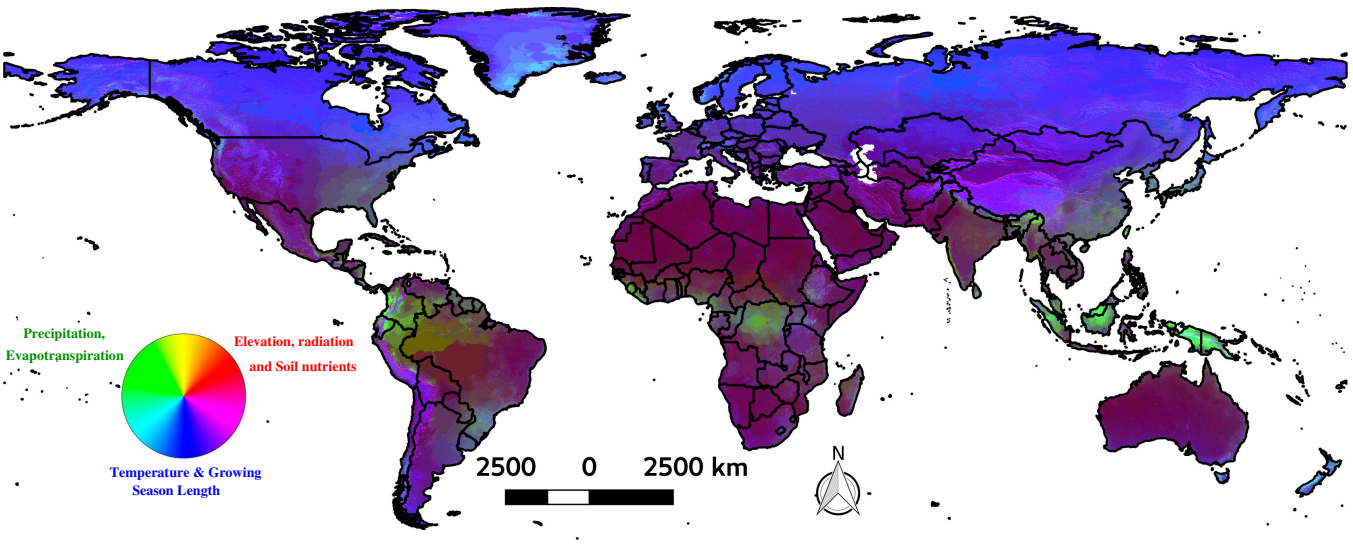

Fig. 11. 1000 Global climate regimes generated by the k-means clustering algorithm for predicted future 2100 by HadCM3 climate model under A1FI emissions scenario. *Similarity color* scheme was used where Red color channel highlights effect of topography and soil properties, Green channel highlight precipitation variables and evapotranspiration, and Blue channel demonstrate the effect of temperature variables and growing season length.

REFERENCES

[1] J. M. Omernik, "Ecoregions of the conterminous united states," *Annals of the Association of American Geographers*, vol. 77, no. 1, pp. 118–125, 1987.

[2] B. Baker, H. Diaz, W. Hargrove, and F. Hoffman, "Use of the kppen-trewartha climate classification to evaluate climatic refugia in statistically derived ecoregions for the people's republic of china," *Climate Change*, vol. 98, pp. 113–131, 2010.

[3] B. M. Steele, "Combining multiple classifiers: An application using spatial and remotely sensed information for land cover type mapping," *Remote Sensing of Environment*, vol. 74, no. 3, pp. 545 – 556, 2000. [Online]. Available: //www.sciencedirect.com/science/article/pii/S0034425700001450

[4] J. Kumar, J. Weiner, W. W. Hargrove, S. P. Norman, F. M. Hoffman, and D. Newcomb, "Characterization and classification of vegetation canopy structure and distribution within the Great Smoky Mountains National Park using LiDAR," in *Proceedings of the 15th IEEE International Conference on Data Mining Workshops (ICDMW 2015)*, P. Cui, J. Dy, C. Aggarwal, Z.-H. Zhou, A. Tuzhilin, H. Xiong, and X. Wu, Eds., Institute of Electrical and Electronics Engineers (IEEE). Conference Publishing Services (CPS), Nov. 2015, pp. 1478–1485.

[5] A. Guisan and W. Thuiller, "Predicting species distribution: offering more than simple habitat models," *Ecology Letters*, vol. 8, no. 9, pp. 993–1009, 2005. [Online]. Available: http://dx.doi.org/10.1111/j.1461-0248.2005.00792.x

[6] W. W. Hargrove and F. M. Hoffman, "Potential of multivariate quantitative methods for delineation and visualization of ecoregions," vol. 34, no. Supplement 1, pp. S39–S60, Apr. 2004.

[7] D. R. Cutler, J. Thomas C. Edwards, K. H. Beard, A. Cutler, K. T. Hess, J. Gibson, and J. J. Lawler, "Random forests for classification in ecology," *Ecology*, vol. 88, no. 11, pp. 2783–2792, 2007.

[8] F. M. Hoffman, J. W. Larson, R. T. Mills, B.-G. J. Brooks, A. R. Ganguly, W. W. Hargrove, J. Huang, J. Kumar, and R. R. Vatsavai, "Data Mining in Earth System Science (DMESS 2011)," in *Proceedings of the International Conference on Computational Science (ICCS 2011)*, M. Sato, S. Matsuoka, P. M. Sloot, G. D. van Albada, and J. Dongarra, Eds., vol. 4. Amsterdam: Elsevier, Jun. 2011, pp. 1450–1455.

[9] F. M. Hoffman and W. W. Hargrove, "Multivariate geographic clustering using a Beowulf-style parallel computer," in *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA '99)*, H. R. Arabnia, Ed., vol. III. CSREA Press, Jun. 1999, pp. 1292–1298.

[10] F. M. Hoffman, W. W. Hargrove, R. T. Mills, S. Mahajan, D. J. Erickson, and R. J. Oglesby, "Multivariate Spatio-Temporal Clustering (MSTC) as a data mining tool for environmental applications," in *Proceedings of the iEMSs Fourth Biennial Meeting: International Congress on Environmental Modelling and Software Society (iEMSs 2008)*, M. Sànchez-Marrè, J. Béjar, J. Comas, A. E. Rizzoli, and G. Guariso, Eds., Jul. 2008, pp. 1774–1781.

[11] J. Kumar, R. T. Mills, F. M. Hoffman, and W. W. Hargrove, "Parallel k-means clustering for quantitative ecoregion delineation using large data sets," in *Proceedings of the International Conference on Computational Science (ICCS 2011)*, M. Sato, S. Matsuoka, P. M. Sloot, G. D. van Albada, and J. Dongarra, Eds., vol. 4. Amsterdam: Elsevier, Jun. 2011, pp. 1602–1611.

[12] R. M. Esteves, T. Hacker, and C. Rong, "A new approach for accurate distributed cluster analysis for big data: competitive k-means," *International Journal of Big Data Intelligence*, vol. 1 (1-2), 2014.

[13] ——, "Competitive k-means, a new accurate and distributed k-means algorithm for large datasets," in *2013 IEEE 5th International Conference on Cloud Computing Technology and Science*, vol. 1, Dec 2013, pp. 17–24.

[14] C. M. Potera, M. C. Mihescu, and M. Mocanu, "An optimized version of the k-means clustering algorithm," in *2014 Federated Conference on Computer Science and Information Systems*, Sept 2014, pp. 695–699.

[15] R. Farivar, D. Rebolledo, E. Chan, and R. H. Campbell, "A parallel implementation of k-means clustering on gpus." in *Pdpta*, vol. 13, no. 2, 2008, pp. 212–312.

[16] S. A. Shalom, M. Dash, and M. Tue, "Efficient k-means clustering using accelerated graphics processors," in *International Conference on Data Warehousing and Knowledge Discovery*. Springer, 2008, pp. 166–175.

[17] P. Mackey and R. R. Lewis, "Parallel k-means++ for multiple shared-memory architectures," in *Parallel Processing (ICPP), 2016 45th International Conference on*. IEEE, 2016, pp. 93–102.

[18] T. Jordan, M. Madden, B. Yang, J. Sharma, and S. Panda, "Acquisition of LiDAR for the Tennessee Portion of Great Smoky Mountains National Park and the Foothills Parkway," Center for Remote Sensing and Mapping Science (CRMS), Department of Geography, The University of Georgia, Athens, Georgia, USA, Tech. Rep. USGS Contract # G10AC0015, 2011.

[19] R. J. Hijmans, S. E. Cameron, J. L. Parra, P. G. Jones, and A. Jarvis, "Very high resolution interpolated climate surfaces for global land areas," *International Journal of Climatology*, vol. 25, no. 15, pp. 1965–1978, 2005. [Online]. Available: http://dx.doi.org/10.1002/joc.1276

[20] E. Saxon, B. Baker, W. Hargrove, F. Hoffman, and C. Zganjar, "Mapping environments at risk under different global climate change scenarios," vol. 8, no. 1, pp. 53–60, Jan. 2005.

[21] B. Baker, H. Diaz, W. Hargrove, and F. Hoffman, "Use of the Köppen-Trewartha climate classification to evaluate climatic refugia in statisti-

cally derived ecoregions for the People's Republic of China," vol. 98, no. 1, pp. 113–131, Jan. 2010.

[22] S. J. Phillips, "Reducing the computation time of isodata and k-means unsupervised classification algorithms," in *Geoscience and Remote Sensing Symposium, 2002 (IGARSS'02)*, vol. 3, Jun. 2002, pp. 1627–1629.

[23] ——, "Acceleration of k-means and related clustering algorithms," in *ALENEX '02: Revised Papers from the 4th International Workshop on Algorithm Engineering and Experiments*, D. M. Mount and C. Stein, Eds. London, UK: Springer-Verlag, 2002, pp. 166–177.

[24] S. Sreepathi, M. L. Grodowitz, R. Lim, P. Taffet, P. C. Roth, J. Meredith, S. Lee, D. Li, and J. Vetter, "Application Characterization Using Oxbow Toolkit and PADS Infrastructure," in *Proceedings of the 1st International Workshop on Hardware-Software Co-Design for High Performance Computing*, ser. Co-HPC '14. IEEE Press, 2014, pp. 55–63. [Online]. Available: http://dx.doi.org/10.1109/Co-HPC.2014.11

[25] C.-K. Luk, R. Cohn, R. Muth, H. Patil, A. Klauser, G. Lowney, S. Wallace, V. J. Reddi, and K. Hazelwood, "Pin: Building customized program analysis tools with dynamic instrumentation," in *Proceedings of the 2005 ACM SIGPLAN Conference on Programming Language Design and Implementation*, ser. PLDI '05. New York, NY, USA: ACM, 2005, pp. 190–200. [Online]. Available: http://doi.acm.org/10.1145/1065010.1065034

[26] J. S. Vetter and M. O. McCracken, "Statistical scalability analysis of communication operations in distributed applications," in *ACM SIGPLAN Symp. on Principles and Practice of Parallel Programming (PPOPP)*. Snowbird, UT: ACM, 2001.

[27] "cuBLAS - NVIDIA's BLAS implementation on top of the CUDA runtime," http://docs.nvidia.com/cuda/cublas/index.html, 2017.

[28] "Titan - Cray XK7 Supercomputer at Oak Ridge National Laboratory," https://www.olcf.ornl.gov/computing-resources/titan-cray-xk7/, 2017.

[29] "TOP500 - Top 500 Supercomputer Sites in the World - June 2015," http://top500.org/lists/2016/11/, 2017.