Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 4 (2011) 1602–1611

Procedia Computer Science

International Conference on Computational Science, ICCS 2011

Parallel *k*-Means Clustering for Quantitative Ecoregion Delineation Using Large Data Sets

Jitendra Kumar[a,1], Richard T. Mills[a], Forrest M. Hoffman[a], William W. Hargrove[b]

[a] *Computer Science and Mathematics Division, Oak Ridge National Laboratory, Oak Ridge, TN, USA*
[b] *Southern Research Station, USDA Forest Service, Asheville, NC, USA*

Abstract

Identification of geographic ecoregions has long been of interest to environmental scientists and ecologists for identifying regions of similar ecological and environmental conditions. Such classifications are important for predicting suitable species ranges, for stratification of ecological samples, and to help prioritize habitat preservation and remediation efforts. Hargrove and Hoffman [1, 2] have developed geographical spatio-temporal clustering algorithms and codes and have successfully applied them to a variety of environmental science domains, including ecological regionalization; environmental monitoring network design; analysis of satellite-, airborne-, and ground-based remote sensing, and climate model-model and model-measurement intercomparison. With the advances in state-of-the-art satellite remote sensing and climate models, observations and model outputs are available at increasingly high spatial and temporal resolutions. Long time series of these high resolution datasets are extremely large in size and growing. Analysis and knowledge extraction from these large datasets are not just algorithmic and ecological problems, but also pose a complex computational problem. This paper focuses on the development of a massively parallel multivariate geographical spatio-temporal clustering code for analysis of very large datasets using tens of thousands processors on one of the fastest supercomputers in the world.

Keywords:
ecoregionalization, *k*-means clustering, data mining, high performance computing

1. Introduction

There has been a long history of both theoretical and utilitarian interest in ecological regions, or ecoregions: regions on a map within which there exist generally similar combinations of ecologically relevant conditions like temperature, precipitation, and soil characteristics. Because species assemblages show a high degree of fidelity to environmental conditions, particular flora and fauna can be strongly associated with particular ecological regions, and delineating the extent of the ecoregions can also be used to infer the the location and size of areas that are potentially suitable for particular types of animals and plants. Thus, the classification of areas into ecoregions is useful for predicting suitable species ranges, for stratification of ecological samples, and to help prioritize habitat preservation and

Email addresses: jkumar@climatemodeling.org (Jitendra Kumar), rmills@climate.ornl.gov (Richard T. Mills), forrest@climatemodeling.org (Forrest M. Hoffman), hnw@geobabble.org (William W. Hargrove)
[1] Corresponding author

1877–0509 © 2011 Published by Elsevier Ltd. Selection and/or peer-review under responsibility of Prof. Mitsuhisa Sato and Prof. Satoshi Matsuoka
doi:10.1016/j.procs.2011.04.173

remediation efforts.

Ecoregions are also used in a more specific way to estimate the extent of suitable habitat for a particular species of interest. In such cases, the ecological habitat requirements of a single species, sometimes referred to as its ecological niche or permissible environmental envelope, are used to identify one or more ecoregions along a gradient of suitability. In the case of threatened or endangered species, a well-executed ecoregion classification can be used to identify and locate the extent of suitable habitat for the purposes of preserving or improving it. In the case of invasive species, an ecoregion classification can be used to gauge the susceptibility of new areas to invasion. Efforts to slow or stop the invasion can be focused within highly suitable areas rather than being squandered on areas that are not as susceptible.

Ecoregions can also be used to extrapolate or interpolate results obtained from a network of measurements or samples. A network in this case is a spatial constellation of sample locations, field stations, or monitoring instruments. Measurements taken within one ecoregion are probably representative of the entire region, and the degree to which another, unmeasured region is similar to the one having a measurement defines how applicable that measurement might be within the unmeasured region. One can also determine how adequate the network of samples itself is for representing the ecological conditions within the greater map that contains it. Regions most poorly represented by the network (i.e., most different from the most similar sample) are the best candidates within which new, additional samples should be taken.

Until recently, ecoregions were usually delineated freehand by human experts, and thus represented codifications of expert opinion. While experts can justify their placement of ecoregion boundaries in particular locations, the methods used in such delineations are neither transparent nor repeatable. Neither are the datasets being used open for inspection. In an effort to make the ecoregion delineation process more rigorously quantitative, two of us [1] began to explore the use of multivariate statistical clustering, using the k-means algorithm, as an approach for ecoregion classification. Such an approach, based on selected ecological characteristics supplied as map layers within a Geographical Information System (GIS) system, quantifies ecological similarity as the inverse of Euclidean distance separating two points in a data space having as many dimensions as there are input map layers, once the normalized values from each map layer are used as coordinates to locate each point. Two points that lie close to each other in data space will, by definition, have a very similar combination of input characteristics, and should perhaps therefore be classified into the same ecoregion. Because the k-means clustering analysis is entirely independent of the geographic location, the process is free to identify even the points widely separated in space as being in the same ecoregion, given enough similarity in the ecological conditions at each point. Thus, even spatially disjoint or distant mountaintops or swamps can be identified as having a high degree of ecological similarity.

A rich body of research in computational data mining exists in the literature with work done by various researchers on the development of clustering algorithms [3, 4, 5, 6]. However, clustering of large datasets remains a computationally intensive and challenging task. Judd et al. [7] designed and implemented a parallel clustering algorithm for a network of workstations. They used a client-server approach where a block of work assignments were sent to each client process, which then calculates the block partial sum of each cluster and sends the results back to the server. The server collects the partial sums from all clients, calculates the new centroids and returns the new centroids to all clients to begin a new iteration. They used Parallel Virtual Machine [8] and Message Passing Interface (MPI) [9] for their implementation. They applied the method to datasets with up to 262,144 records and tested the scaling for up to 16 processors. Dhillon and Modha [10] developed a parallel k-means algorithm based on a message-passing model. They distributed the data to each parallel process to operate on them independently. A synchronization step was carried out at the end of each iteration using `MPI_Allreduce`. They applied their implementation to datasets with up to 2,097,152 records (with 16 dimensions per record) and achieved excellent scaling for up to 16 processors. Li and Chung [11] developed a parallel bisecting k-means algorithm and implemented it in Java, using Remote Method Invocation (RMI) for interprocess communication. The largest dataset tested by them had 2,097,152 records (with 8 dimensions per record). They achieved good scaling with their method up to 8 processors on a Linux cluster. While design and implementation of parallel clustering algorithms has been reported by many researchers in the literature, the application of most of them has been limited to small datasets on small scale computing clusters. State-of-the-art satellites, airborne and ground-based remote sensing stations and global climate models provide massive amounts of

data that can be utilized for ecology and environmental sciences studies. The focus of work reported here is the design and implementation of a massively parallel clustering algorithm to enable the analysis of massive amounts of data using some of the largest supercomputers in the world.

2. Cluster analysis algorithm

2.1. The k-means Algorithm

k-means is a popular algorithm for clustering of a dataset (X_1, X_2, ..., X_n) with n records into k clusters. The number of desired clusters, k, is supplied as an input to the algorithm and remains fixed. The iterative algorithm starts with initial "seed" centroids (C_1, C_2, ..., C_k) and tests the Euclidean distance of each map cell (X_i, $1 \leq i \leq n$) to every centroid (C_j, $1 \leq j \leq k$), classifying it to the closest existing centroid. After all map cells are classified, a new centroid is calculated as the mean of all dimensions of each map cell classified to that centroid. Thus, the centroids move through the data space while the map cells remain fixed. The classification converges and iterations stop when fewer than a fixed number of map cells change their cluster assignment from the previous iteration (we use $< 0.05\%$).

2.2. Speedup Using the Triangle Inequality

Ecoregion classifications increase in utility and sophistication as they cover wider areas with higher spatial resolution, and consider greater numbers of ecological characteristics. The highly multivariate nature of the problem, combined with the exponential increase in map cells with increasing resolution, leads to large datasets and significant computational effort, necessitating a parallel computational approach. Unlike many parallel problems, however, the size of the input data sets can make the k-means algorithm more I/O limited than CPU-bound. Nevertheless, the great majority of the computations consist of Euclidean distance calculations and comparisons in higher-order data space.

An acceleration technique using the triangle inequality [12, 13] was implemented that reduces the number of Euclidean distance calculations by establishing an upper limit on distances between data points and centroids. The speedup algorithm eliminates unnecessary point-to-centroid distance calculations and comparisons based on the previous cluster assignment and the new inter-centroid distances. Using the triangle inequality [12], if the distance between the last centroid assignment C_l and the new candidate centroid C_m is greater than or equal to the distance between the point X_i and the last centroid C_l, then calculation of distance between the point X_i and the candidate centroid C_m is not required. The triangular inequality states that $d(C_l, C_m) \leq d(X_i, C_l) + d(X_i, C_m)$ where the function $d(a, b)$ represents the Euclidean distance between the points a and b. Thus, $d(X_i, M_l) \geq d(M_l, M_m) - d(X_I, C_l)$. Therefore, if $d(C_l, C_m) \geq 2d(X_i, C_l)$, it can be concluded that $d(X_i, C_m) \geq d(X_i, C_l)$ without calculating $d(X_i, C_m)$. Thus the candidate centroid, C_m, can be eliminated without ever computing its distance to the point X_i because the last centroid, C_l, is equally close or closer. The required distance evaluations can be further reduced by sorting the inter-centroid distances. The new candidate centroids are evaluated in the order of their distance from the former centroid, C_l. No evaluations are required once the critical distance $2d(X_i, C_l)$ is surpassed. The nearest centroid is known from a previous evaluation. When the number or points, n, is much larger than the number of clusters, k, the cost of sorting inter-centroid distances is negligible. The time required to complete each iteration also decreases as the clusters approach convergence, *i.e.*, fewer data points change their cluster membership, and the locations of the cluster centroids stabilize.

2.3. Empty Clusters and Centroid Warping

It is possible for a centroid to have no member map cells assigned to it at the end of an iteration. Such empty clusters, if not populated at convergence, will result in dividing the map into fewer ecoregions than specified by the user. Our implementation handles empty clusters using a "worst of the worst" method. During the cluster assignment process, the sum of distances of all the points in a cluster to the cluster centroid is calculated at each process. In the event of an empty cluster, a global reduction operation is carried out to identify the "worst" cluster with the highest average distance. Once the "worst" cluster is chosen, the "worst" point in that cluster is identified and is assigned to the empty cluster with its centroid set to be at that point. The centroids of all the clusters are updated at the end of the iteration to reflect the changes due to warping. Each process keeps track of the information required to identify the worst cluster and the point, although a global reduction is performed only when a empty cluster is found. A cluster

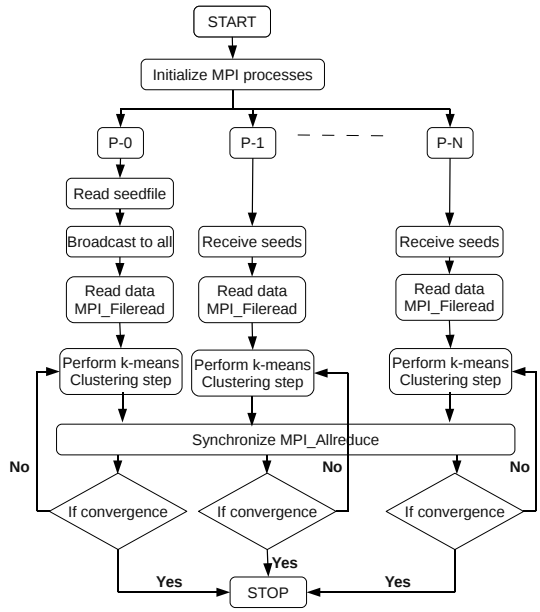


Figure 1: Flowchart for parallel *k*-means algorithm

with a single member is never chosen for warping of an empty cluster. Any number of empty clusters can be handled using this method. Convergence is not reached in an iteration if an empty cluster found.

2.4. Parallel k-means Clustering Implementation

A massively parallel and scalable implementation of a parallel *k*-means algorithm was developed for cluster analysis of very large datasets (Figure 1). The implementation was done in the C language using the Message Passing Interface (MPI) for distributed memory parallelism [9]. The dataset to be clustered is divided among the available number of processes, which read the data from a single file using MPI parallel file I/O. The initial seed file is read by one processor (Rank 0) and is broadcast to all the processes. The algorithm works exactly the same way as the standard serial *k*-means algorithm. However, every process operates only on its chunk of the dataset, carries out the distance calculation of points from the centroids and assigns it to the closest centroid. Each process also calculates the partial sum along each dimension of the points in each cluster for its chunk for the centroid calculation. At the end of the iteration, a global reduction operation is carried out, after each process obtains the information, to calculate the new cluster centroids. Iterations are carried out until convergence, after which the cluster assignments are written to an output file. For portability reasons, the options to use MPI parallel I/O or serial file I/O with scatter/gather operations was implemented. For smaller data sets, use of serial I/O is recommended to avoid the overhead associated with parallel I/O.

3. Application

3.1. Study Datasets

Performance of the parallel *k*-means algorithm and implementation was evaluated using three real-world example datasets. These three datasets (Table 1) have been used in this study to test a broad combination of small to large n and k clustering problems and to test the computational efficiency and scalability of the implementation.

Table 1: Summary of datasets used

Dataset	No. of dimensions	No. of records	Dataset size
fullUS	25	7,801,710	745 MB
AmeriFlux	30	7,856,224	900 MB
Phenology	22	1,024,767,667	84 GB

fullUS Dataset: The dataset is a national map of the conterminous United States at 1 km^2 resolution (nearly 8M map cells) consisting of 25 variables from maps with values for elevation, soil nitrogen, soil organic matter, soil water capacity, depth to water table, mean precipitation, solar irradiance, degree-day heat sum, and degree-day cold sum.

AmeriFlux Dataset: The second example strives to produce homogeneous national carbon flux ecoregions, starting with 30 ecologically relevant characteristics. The input characteristics include factors relating to the abiotic environment, including temperature and precipitation during the growing and non-growing seasons, and several soil characteristics, as well as satellite-based characteristics of the vegetation, including seasonal leaf area index and greenness, as well as percent tree and shrub cover. Finally, modeled estimates of seasonal gross primary productivity and respiration are included. The proximate purpose of this ecoregionalization was to gauge the adequacy and representativeness of the DOE AmeriFlux network of carbon eddy flux co-variance measurement towers. The input layers and purpose are more thoroughly described in [5] and at `http://www.geobabble.org/flux-ecoregions/`.

Phenology Dataset: The third example sets out to delineate national phenological ecoregions, or phenoregions. Phenoregions are ecoregions within which the vegetation has the same timing of green-up in the spring and brown-down in the fall. This third dataset consists of a temporally-smoothed MODIS-based greenness index, calculated and accumulated at 16-day intervals for the calendar year, over the years 2003 through 2009 nationally at 231 m resolution. There are 146,395,381 map cells in the phenology dataset, each with 22 characteristics, consisting of consecutive greenness observations. The dataset is divided into homogeneous phenoregions nationally.

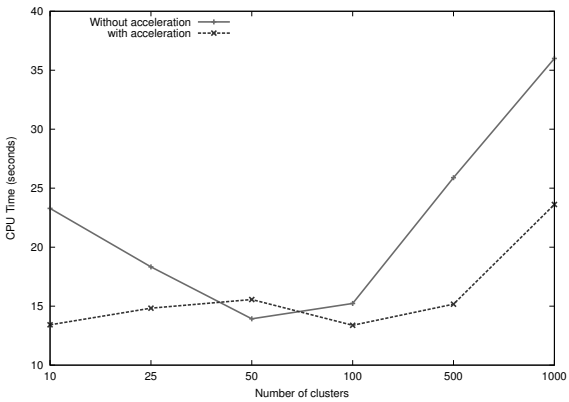
3.2. Computational resources used

Work reported in this study was carried out using Jaguar, a Cray XT5 supercomputer at Oak Ridge National Laboratory. It consists of 18,688 compute nodes of dual hex-core AMD Opteron 2435 (Istanbul) processors running at 2.6 GHz, 16 GB of DDR2-800 memory, and a SeaStar 2+ router. It contains a total of 224,256 processing cores, 300 TB of memory, and a peak performance of 2.3 petaflops.

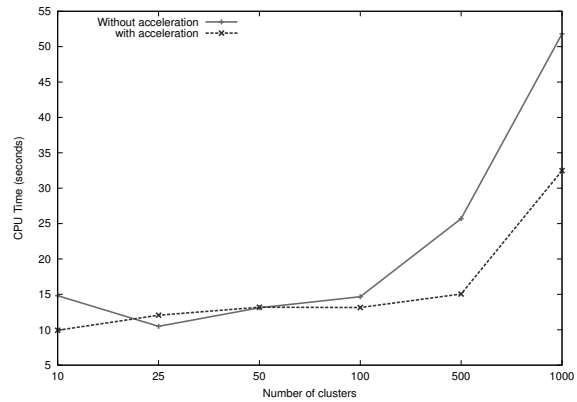
4. Results and discussions

4.1. Scaling with increasing k (number of clusters) and n (dataset size)

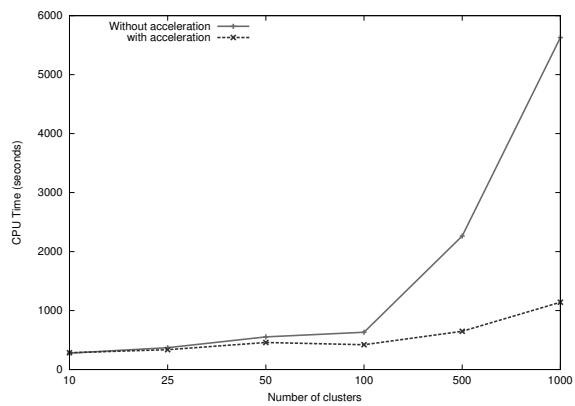
The parallel k-means algorithm was applied to three above mentioned datasets of different sizes (n) and for different number of desired clusters (k). The purpose of these studies was to understand the performance and scalability of the algorithm for clustering increasingly large datasets and for increasing numbers of clusters. Figure 2 shows the scaling of the three datasets of increasing size for increasing numbers of clusters, k = 10, 25, 50, 100, 500 and 1000. Good performance was achieved by the algorithm in terms of simulation time for all the datasets. Also, a significant improvement in the scaling was achieved by the use of the triangle inequality acceleration technique. As the required number of distance evaluation increases with the increase in the number of desired clusters, the improvement due to use of acceleration techniques increases significantly. Figure 3 shows the number of distance calculations required for the clustering with and without the use of triangle inequality acceleration. Simulations for results presented in Figures 2 and 3 were carried out using 1024 processing cores.



(a) fullUS dataset: with and without acceleration

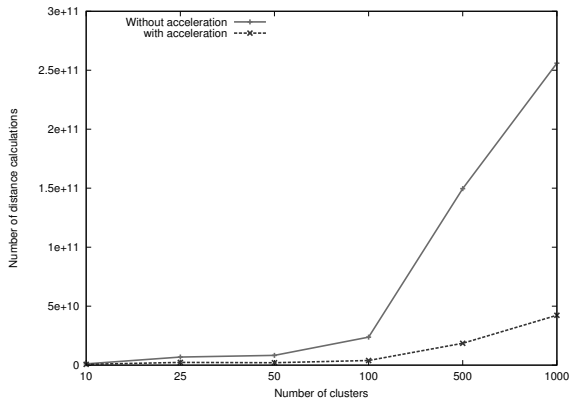


(b) AmeriFluxIIIA dataset: with and without acceleration

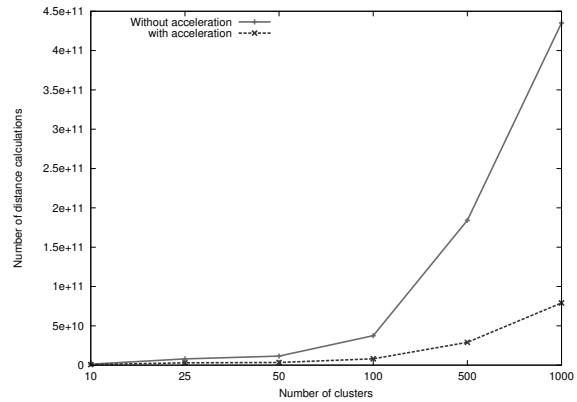


(c) Phenology dataset: with and without acceleration

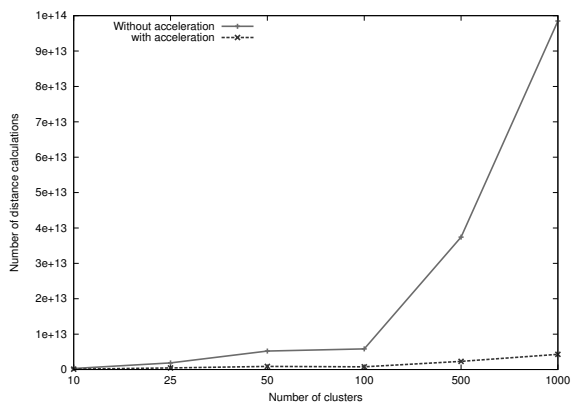
Figure 2: Simulation time of parallel k-means algorithm with increasing k and n (X-Axis: Number of desired clusters, Y-Axis: CPU Time)



(a) fullUS dataset: with and without acceleration

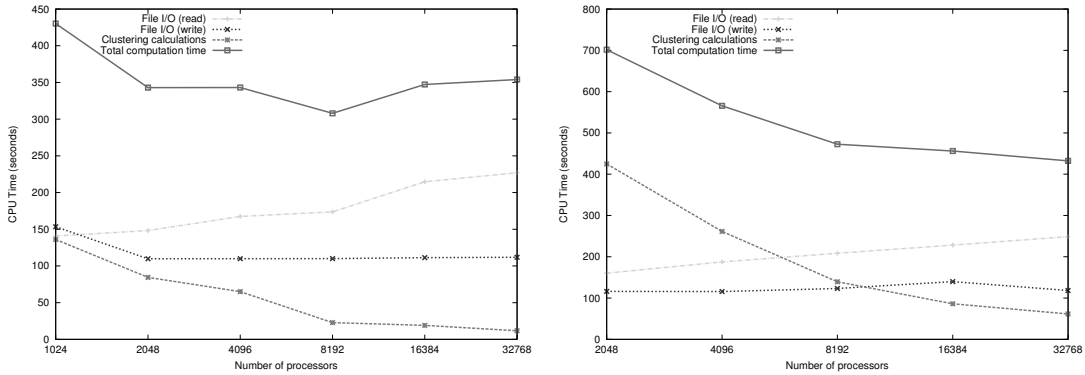


(b) AmeriFluxIIIA dataset: with and without acceleration



(c) Phenology dataset: with and without acceleration

Figure 3: Scaling with increasing k and n (X-Axis: Number of desired clusters, Y-Axis: Number if Euclid distance calculations)



(a) No. of clusters (k) = 50 (b) No. of clusters (k) = 1000

Figure 4: Scaling of the parallel k-means algorithm on Jaguar Cray XT5 for the Phenology dataset

4.2. Scaling with processing cores

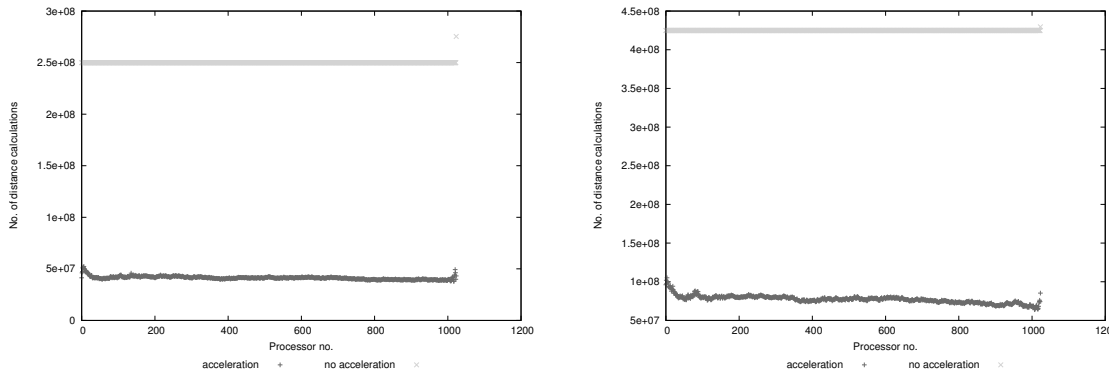
Computational complexity of the clustering simulations increases with the increase in the size of the dataset and the number of desired clusters leading to a very long time to solution. Large numbers of processors available on ultra-scale supercomputers can be used to reduce the time to solution for the clustering simulations and make it feasible to carry out large numbers of simulations required for environmental and ecological studies. However, it is necessary to design the implementation to efficiently use larger numbers of processors. A detailed scaling analysis of the developed parallel k-means algorithm was carried out. Figure 4 shows results of the scaling analysis of the algorithm using the Phenology dataset for 50 and 1000 desired clusters. Excellent scaling was achieved up to 32,768 processors on the Jaguar Cray XT5 system for the dataset of approximately 84 GB in size (single precision binary). It can be observed that the time spent doing file I/O (read and write) during the cluster analysis increases slightly with the increase in number of processors, but the time for distance calculations is significantly reduced with computations being distributed to a larger number of processors, leading to a reduction in time-to-solution. The increase in file I/O times (especially read) with increasing processing cores is an artifact of Lustre filesystem performance on the Cray XT5 system. For 50 clusters (Figure 4a) a decrease in simulation time is observed up to 8192 processors, until file I/O overhead takes over the gain in speedup due to parallel distance calculations by the algorithm, leading to an overall increase in simulation time. Similar patterns of file I/O overhead (Figure 4b) were also observed for the 1000 clusters case. For larger numbers of desired clusters requiring significantly more distance calculations, better speed up is achieved by the parallel clustering algorithm. The time spent in file I/O is negligible compared to the speedup achieved by the distance calculations, thus good scaling is maintained up to 32,768 processors (Figure 4b). The developed implementation can be readily applied to analysis of even larger datasets and can efficiently use hundreds of thousands processors on modern supercomputers.

4.3. Load imbalance

For the standard k-means algorithm, with an equal distribution of data among the parallel processes, exactly the same amount of computation would be required by each process for clustering. However, use of the triangle inequality acceleration technique avoids many unnecessary distance evaluations. This reduction in computation will usually be different for different processes, inducing some load imbalance in the parallel cluster algorithm. The magnitude of load imbalance depends on the dataset and on the number of clusters desired. Figure 5 shows the load imbalance in terms of the number of distance calculations performed by each process (using 1024 processes) for the standard k-means and triangle inequality accelerated parallel k-means algorithms for the fullUS and AmeriFlux datasets. While some load imbalance is introduced by the triangle inequality acceleration algorithm, it is acceptable given the significant reduction in the computation achieved by it.

4.4. Ecoregionalization based on the Phenology dataset

A wide range of cluster simulations were carried out for all three datasets, and results were analyzed. Presented here are some results and analysis based on the phenology dataset. Figure 6 shows a map of phenoregions for the



(a) fullUS dataset: with and without acceleration (b) AmeriFluxIIIA dataset: with and without acceleration

Figure 5: Load imbalance due to triangle inequality acceleration

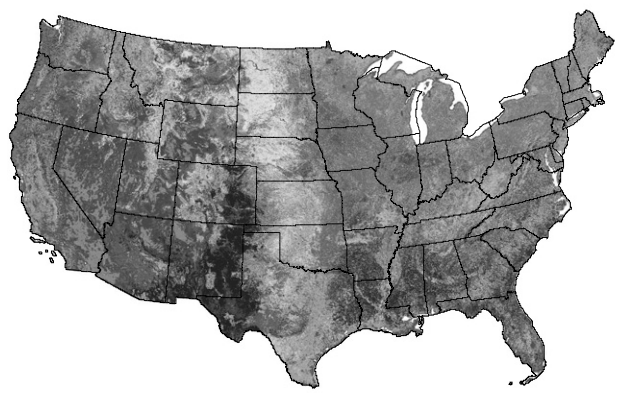


Figure 6: The 2008 map of 50 phenoregions defined for the CONUS derived from cluster analysis of Phenology data

conterminous United States for the year 2008 derived from geospatiotemporal cluster analysis from the Phenology dataset. Comparison of phenostates over time could allow the identification of abnormal events, like change in vegetation health, forest fires, disease outbreaks, etc. Such analyses are one of the primary motivation for the current work under the U.S. Forest Service's Forest Incidence Recognition and State Tracking (FIRST) program [14]. The results of the cluster analysis of Phenology datasets has been successfully applied to the identification of several anomalous forest incidents, like the mountain pine beetle (MPB) outbreak in Colorado, USA. Figure 7 shows the results of such an analysis of phenostate transition between 2003 and 2008 in Colorado successfully identifying the MPB outbreak.

5. Conclusion

Such efforts as these should make it possible to realistically consider clustering even larger datasets, such as the entire historical run of MODIS data, which has covered the conterminous United States at 500 m^2 resolution every 16 days, from 2002 to present. We envision that such a dataset, in conjunction with parallel clustering implementations such as those described here, could form the basis for a national ecological monitoring and early warning system. This system could compare the state of ecosystems as just observed to all past historical states, checking for deviations from expected or nominal seasonal behavior. Unusual or unexpected activity, once highlighted, could be verified on the ground for the presence of ecological threats like wildfire, insect outbreaks, or invasive species. Once the exhaustive historical clustered states are statistically identified, the assignment of each map cell to the most similar pre-defined

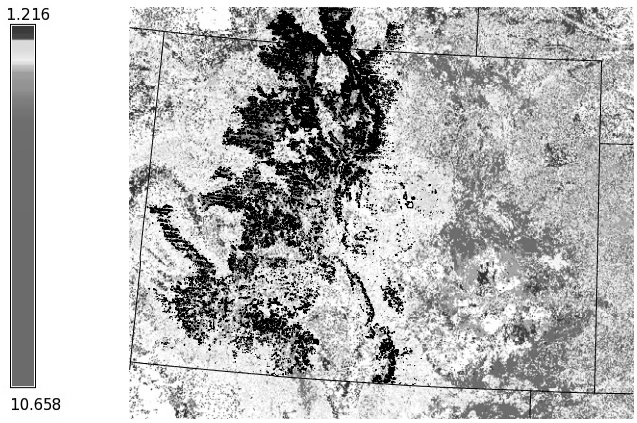


Figure 7: The map of relative transition of phenoregions between 2003–2008 for Colorado, USA

state is not computationally difficult. However, the successful ingestion and clustering of a long remote sensing record like MODIS, in concert with actual climatic, soil, and topographic data, will require an extremely efficient parallel implementation of multivariate clustering.

6. Acknowledgments

This research was partially sponsored by the U.S. Department of Agriculture, U.S. Forest Service, Eastern Forest Environmental Threat Assessment Center. This research used resources of the National Center for Computational Science at Oak Ridge National Laboratory, which is managed by UT-Battelle, LLC, for the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

7. References

[1] W. W. Hargrove, F. M. Hoffman, Using multivariate clustering to characterize ecoregion borders, Computing in Science & Engineering 1(4) (1999) 18–25.
[2] W. W. Hargrove, J. P. Spruce, G. E. Gasser, F. M. Hoffman, Toward a national early warning system for forest disturbances using remotely sensed canopy phenology, Photogrammetric Engineering & Remote Sensing 75(10) (2009) 1150–1156.
[3] J. Hartigan, Clustering Algorithms, John Wiley & Sons, 1975.
[4] F. M. Hoffman, W. W. Hargrove, Multivariate geographic clustering using a Beowulf-style parallel computer, in: H. R. Arabnia (Ed.), Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA '99), ISBN 1-892512-11-4, Vol. III, CSREA Press, 1999, pp. 1292–1298.
[5] W. W. Hargrove, F. M. Hoffman, Potential of multivariate quantitative methods for delineation and visualization of ecoregions, Environmental Management 34(5) (2004) s39–s60.
[6] A. K. Jain, M. N. Murty, P. J. Flynn, Data clustering: a review, ACM Comput. Surv. 31 (1999) 264–323.
[7] D. Judd, P. McKinley, A. Jain, Large-scale parallel data clustering, Pattern Analysis and Machine Intelligence, IEEE Transactions on 20 (8) (1998) 871 –876.
[8] V. S. Sunderam, PVM: a framework for parallel distributed computing, Concurrency: Pract. Exper. 2 (1990) 315–339.
URL `http://portal.acm.org/citation.cfm?id=95324.95327`
[9] W. Gropp, E. Lusk, A. Skjellum, Using MPI: Portable Parallel Programming with the Message Passing Interface, The MIT Press, Cambridge, MA, 1996.
[10] I. Dhillon, D. Modha, A data-clustering algorithm on distributed memory multiprocessors, in: M. Zaki, C.-T. Ho (Eds.), Large-Scale Parallel Data Mining, Vol. 1759 of Lecture Notes in Computer Science, Springer Berlin / Heidelberg, 2000, pp. 802–802.
[11] Y. Li, S. Chung, Parallel bisecting k-means with prediction clustering algorithm, The Journal of Supercomputing 39 (2007) 19–37, 10.1007/s11227-006-0002-7.
[12] 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, 2002, pp. 1627–1629.
[13] S. J. Phillips, Acceleration of k-means and related clustering algorithms, in: D. M. Mount, C. Stein (Eds.), ALENEX '02: Revised Papers from the 4th International Workshop on Algorithm Engineering and Experiments, Springer-Verlag, London, UK, 2002, pp. 166–177.
[14] F. M. Hoffman, R. T. Mills, J. Kumar, S. S. Vulli, W. W. Hargrove, Geospatiotemporal data mining in an early warning system for forest threats in the United States, in: Proceedings of the 2010 IEEE International Geoscience and Remote Sensing Symposium (IGARSS 2010), Honolulu, Hawaii (Invited), 2010.