Argonne NATIONAL LABORATORY

Scalable Geospatiotemporal Clustering on Novel Fine-Grained Parallel Computer Architectures

Richard Tran Mills, Argonne National Laboratory; Vamsi Sripathi, Intel Corporation; Sarat Sreepathi, Oak Ridge National Laboratory; Jitendra Kumar, Oak Ridge National Laboratory, Forrest M. Hoffman, Oak Ridge National Laboratory; William W. Hargrove, USDA Forest Service Southern Research Station

US-IALE Annual Meeting, Chicago, IL

April 11, 2018

Introduction

- Increasing availability of high-resolution geospatiotemporal data sets from varied sources:
  - Observatory networks
  - Remote sensing platforms
  - Computational Earth system models
- Open new possibilities for knowledge discovery and mining of ecological data sets fused from disparate sources.
- Traditional algorithms/computing platforms impractical for analysis/synthesis of data sets this large: Need new approaches to utilize complex memory hierarchies and high levels of available parallelism in state-of-the-art high-performance computing platforms.
- We have adapted pKluster—an open-source tool for accelerated k-means clustering we use for many geospatiotemporal applications—to effectively utilize state-of-the art multi- and manycore processors, such as the second-generation Intel Xeon Phi ("Knights Landing") processor, as well as GPGPUs.

Scalable k-means Clustering with pKluster

Our distributed-memory clustering code has a long history...

Figure: Originally developed in 1996–1997 for use on the Stone Soupercomputer, a very early Beowulf-style cluster constructed entirely out of surplus parts (see "The Do-It-Yourself Supercomputer", *Scientific American*, 265 (2), pp. 72-79, 2001.)

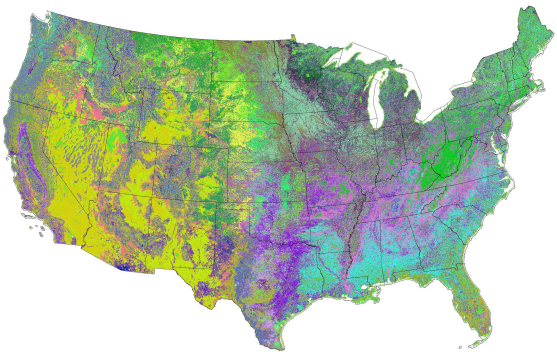
Quantitative Ecoregionalization and Sampling Network Design

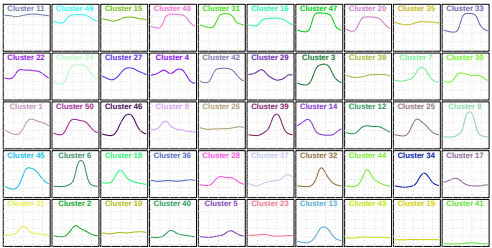
(a) 10 ecoregions, present (2000-2009)

(b) 10 ecoregions, future (2090-2099)

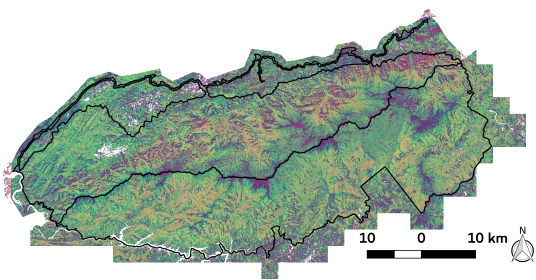
Figure: Geospatiotemporal clustering of a combination of observational data and downscaled general circulation model results projects dramatic shifts in location of Alaska ecoregions using downscaled 4 km GCM results. Arctic tundra projected to be at 0.78% of current extent by 2099. DOI: 10.1007/s10980-013-9902-0. **2014 US-IALE Outstanding Paper in Landscape Ecology.**

MODIS NVDI-based phenoregionalization





GSMNP LiDAR-derived canopy structure classification



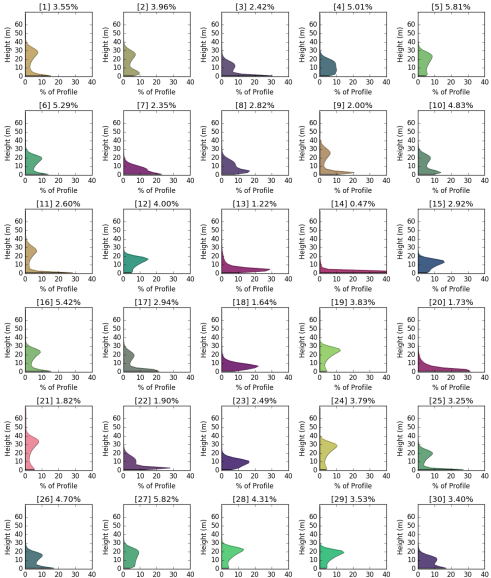


Figure: Map (above) showing the 30 most-different classes of vegetation canopy structure, as identified by k-means clustering (right) for the Great Smoky Mountains National Park.

Scalable *k*-means Clustering with pKluster

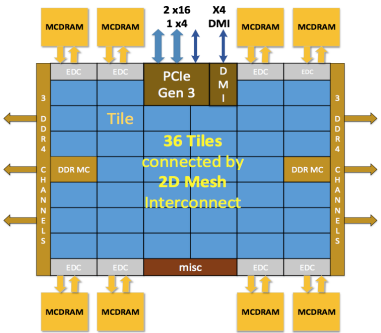
- When pKluster was initially written, on-node parallelism was virtually nonexistent on commodity PCs; the focus was purely on distributed-memory parallelism.
- Because of extreme heterogeneity of the cluster, a master-slave parallel programming paradigm was used (provides dynamic load-balancing).
  - On modern, a fully-distributed, masterless approach may be more efficient.
  - We work with the master-slave version here, because some techniques used here introduce load imbalance even on homogeneous machines.

Features:

- Runs on any machine (or cluster) with C89 (or higher) C compiler and an MPI implementation.
- Option to improve cluster quality by moving or "warping" clusters that become empty to locations in data space where points that are farthest from their current cluster centroids reside.
- Support for clustering observation vectors with many zero entries (e.g., species occurrence data).
- **Fast!** Suitable for clustering multi-terabyte data sets.
  - Implements "accelerated" *k*-means algorithm.
  - Optimizations for manycore CPU and GPGPU systems.

Manycore Computing Architectures

- In recent years, the number of compute cores and hardware threads has been dramatically increasing.
- Seen in GPGPUS, "manycore" processors such as the Intel Xeon Phi, and even on standard server processors (e.g., Intel Xeon Skylake).
- There is also increasing reliance on data parallelism/fine-grained parallelism.
  - Current Intel Xeon processors have 256-bit vector registers and support AVX2 instructions.
  - Second-generation Intel Xeon Phi processors and Intel Skylake Server processors have 512-bit vectors/AVX512 instructions.



At left, "Knights Landing" (KNL) Xeon Phi processor:

- Up to 36 tiles interconnected via 2D mesh
- Tile: 2 cores + 2 VPU/core + 1 MB L2 cache
- Core: Silvermont-based, 4 threads per core, out-of-order execution
- Dual issue; can saturate both VPUs from a single thread
- 512 bit (16 `floats` wide) SIMD lanes, AVX512 vector instructions
- High bandwidth memory (MCDRAM) on package: 490+ GB/s bandwidth on STREAM triad[2]

Benchmarking Platforms and Problem

	Intel(R) Xeon(R) CPU E5-2697 v4	Intel(R) Xeon(R) Gold 6148	Intel(R) Xeon Phi(TM) CPU 7250
Code Name	Broadwell (BDW)	Skylake (SKX)	Knights Landing (KNL)
Sockets	2	2	1
Cores	36	40	68
Threads (HT enabled)	72	80	272
CPU Clock (GHz)	2.3	2.4	1.4
HBM	-	-	16 GB
Memory	128 GB @ 2400 MHz	192 GB @ 2666 MHz	98 GB @ 2400 MHz
ISA	AVX2	AVX512{F, DQ, CD, BW, VL}	AVX512{F,PF, ER, CD}

Benchmark problem: GSMNP LiDAR clustering

- 1.5 million observations
- 74 coordinates
- $k = 2000$ clusters

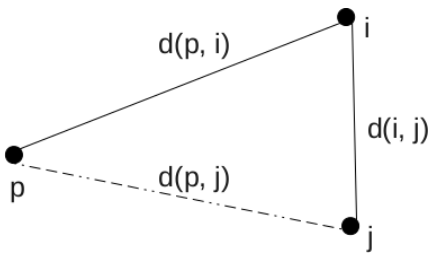
Parallel k-means clustering algorithm

- Centralized master-worker paradigm
- Start from some initial centroids (chosen offline)
- Master:
  - Broadcasts centroids and aliquot assignment to workers
  - Collects new cluster assignments from workers
  - Recomputes centroids
- Workers, for an assigned aliquot:
  - Compute observation-to-centroid distances
  - Assign each observation to closest centroid

Figure: Illustration of k-means iteration for $k = 3$. `https://commons.wikimedia.org/wiki/File:K-means_convergence.gif`

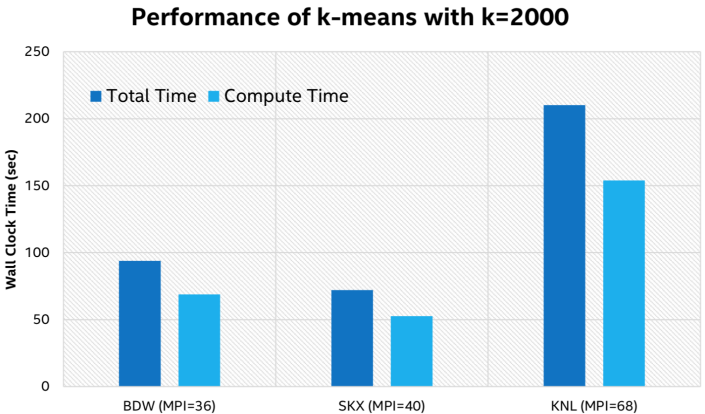
Accelerated k-means clustering

- Classical k-means actually performs more distance calculations than required!
- Use the triangle inequality to eliminate unnecessary point-to-centroid distance computations based on the previous cluster assignments and the new inter-centroid distances.
- Reduce evaluation overhead by sorting inter-centroid distances so that new candidate centroids c_j are evaluated in order of their distance from the former centroid c_i. Once the critical distance $2d(p, c_i)$ is surpassed, no additional evaluations are needed, as the nearest centroid is known from a previous evaluation.



$d(i,j) \leq d(p,i) + d(p,j)$
$d(i,j) - d(p,i) \leq d(p,j)$
if $d(i,j) \geq 2d(p,i)$:
$d(p,j) \geq d(p,i)$
without calculating the distance $d(p,j)$

Baseline Performance

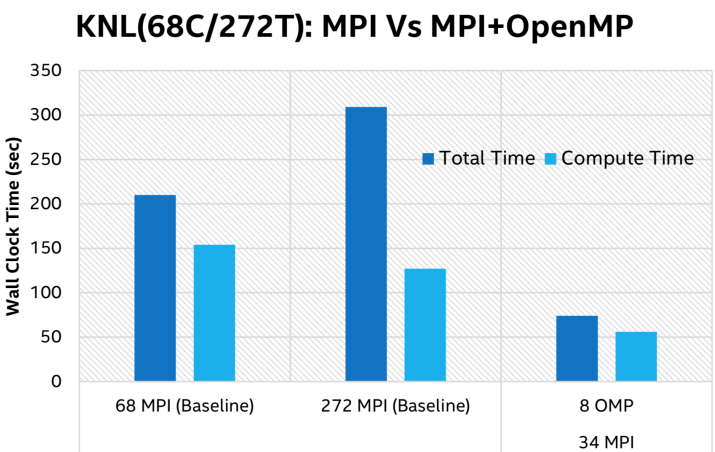


- 1.3X speedup on SKX vs. BDW
- Significant slowdown (2.2X) on KNL

Effective Use of Hyperthreads

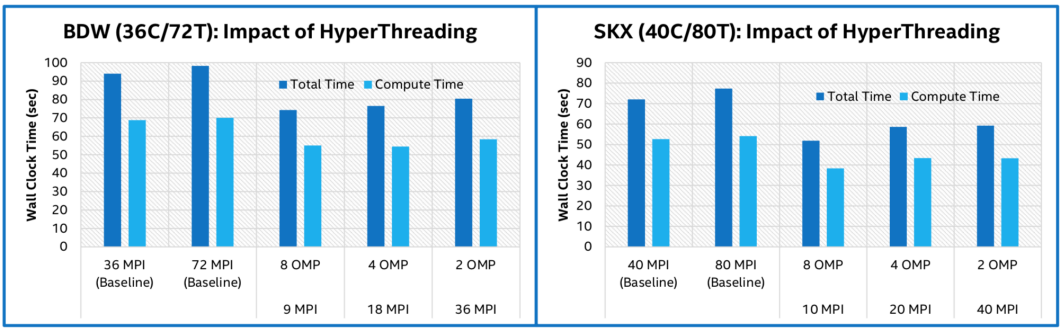
- Using a pure MPI approach (one MPI rank per core), performance of the accelerated k-means clustering approach is surprisingly poor on the "Knights Landing" (KNL) processor.
- Using two MPI ranks per core slightly decreases time in the actual clustering calculation, but slightly increases total time due to greater overhead in master-worker coordination.
- This suggests that using more available hardware threads can improve performance on KNL, if we can avoid increasing master-worker overhead.

Performance Optimizations: OpenMP Parallelism on KNL



- Hybrid MPI-OpenMP version of distance calculation function effectively utilizes FMA units and reduces the bottleneck on rank 0.
- Use dynamic loop scheduling to smooth load imbalance due to triangle inequality (many observations in an aliquot might skip point-to-centroid distance calculation).
- Pin each MPI to a KNL "tile" and spawn 8 threads (4 threads per core).
- 2.8X improvement.

Performance Optimizations: OpenMP Parallelism on BDW and SKX

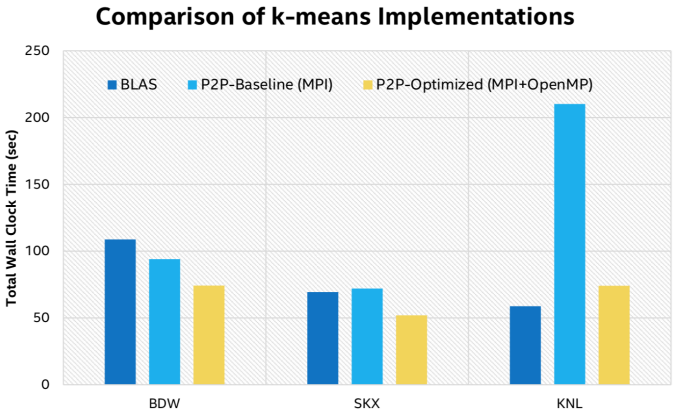


- Hybrid MPI-OpenMP implementation enables to effectively use hyper threads/logical threads
- BDW: 26% improvement with 9 MPI and 8 OMP
- SKX: 38% improvement with 10 MPI and 8 OMP

Improving computational intensity

- Can achieve greater computational intensity of the observation–centroid distance calculations by expressing the calculation in matrix form:
  - For observation vector x_i and centroid vector z_j, the squared distance between them is $D_{ij} = \left\|x_i - z_j\right\|^2$.
  - Via binomial expansion, $D_{ij} = \|x_i\|^2 + \left\|z_j\right\|^2 - 2x_i \cdot z_j$
  - The matrix of squared distances can thus be expressed as $D = \overline{x}\mathbf{1}^\intercal + \mathbf{1}\overline{z}^\intercal - 2X^\intercal Z$, where X and Z are matrices of observations and centroids, respectively, stored in columns, $\overline{x}$ and $\overline{z}$ are vectors of the sum of squares of the columns of X and Z, and $\mathbf{1}$ is a vector of all 1s.
- Above expression can be calculated in terms of a level-3 BLAS operation (xGEMM), followed by two rank-one updates (xGER, a level-2 operation).
- We use highly optimized BLAS implementations from Intel's MKL and NVIDIA cuBLAS to speed up distance calculations on Xeon Phi and GPGPUs, respectively.
- Distance calculations using above formulation can be dramatically faster than the straightforward loop over vector distance calculations when many distance comparisons must be made.
- Using the matrix formulation for distance comparisons in early k-means iterations is straightforward; a more complicated approach we hope to explore is using the matrix formulation in combination with the acceleration techniques described above, in which only a subset of observation–centroid distances are calculated.

Performance Summary



- BLAS formulation provides the best performance on KNL, slightly slower then P2P distance calculation SKX.
- Overall performance improvements:
  - KNL: 3.5X
  - BDW: 1.3X
  - SKX: 1.4X

Future Directions: Software Development

- Investigate hybrid approach combining accelerated k-means method and matrix formulation within the same iteration.
- Re-implement a fully distributed, masterless approach in the current version of the code to handle cases in which master-slave overhead is high (e.g., many cases on `KNL`).
- Add support for emerging high-capacity, non-volatile memory technologies.
- Supported open-source release under Apache License 2.0.

Future Directions: Possible Science Goals

- Potential questions of interest:
  - How are global plant distributions affect by climate change?
  - What are the implications for global carbon budgets and feedbacks to climate?
  - What changes do we expect to key events like onset of growing season?
  - What changes do we expect to suitable growing ranges for crops?
  - Are there policy implications for agriculture and ensuring the food supply?
- Could combine analysis to all of the MODIS vegetative phenology record with global fine-scale meteorological reanalysis and possibly other ancillary data layers.
  - Enables attribution of vegetation changes to climate or other events.
  - Study directly observed vegetation responses to extreme events.
- Could analyze high-resolution and/or multi-model ensemble Earth system model simulations:
  - Project changes to distribution of eco-phenoregions (identified by the historical analysis) for different climate change scenarios.
  - Combine with crop physiology models to project changes in yields.
  - Combine with urban growth models or population models to assess resource planning, policy scenarios, and crop futures.
- Potential collaborators: Beta users of pKluster are welcome! What is **your** scientific question?