Chapter 10

Cloud-enabling dust storm forecasting

Qunying Huang, Jizhe Xia, Manzhu Yu, Karl Benedict, and Myra Bambacus

Real-time dust storm forecasting is a typical application of computing intensities, which requires fast, high-resolution, and large geographic area simulation. This chapter introduces how to use cloud services to support dust storm simulation and predictions.

10.1 DUST STORM MODELING: BACKGROUND AND CHALLENGES

10.1.1 Background

Dust storms result from strong turbulent wind systems entraining dust particles into the air and reducing visibility from miles down to several meters (Goudie and Middleton 1992). Global climate change has driven up the frequency and intensity of dust storms in the past decades with negative consequences on the environment, human health, and physical assets. For example, severe dust storms impact the southwestern United States several times per year and cause major vehicle accidents, property damage, injuries, and loss of lives (Shoemaker and Davis 2008). The negative impacts of dust storms have motivated scientists to develop models to better understand and predict the distribution and intensity of dust emission, deposition, and structure. Since the late 1980s, several research groups have developed dust models that can correctly predict spatiotemporal patterns, evolution, and the magnitude order of dust concentration, emissions, and deposition (Huang et al. 2013a). In addition to weather and climate prediction, dust storm modeling and forecasting could also be used in practice (WMO 2012) to: (1) provide important implications of air quality for constituencies ranging from the public to industry; (2) assist public health services with improved operational disease surveillance and early detection; (3) improve air and highway safety and accident emergency management; and (4) serve as a warning advisory and assessment system to give instructions for adaptive action, such as changing the time of planning, strengthening infrastructure, and construction of windbreaks and shelterbelts.

10.1.2 Challenges

Utilizing existing dust models to predict high-resolution dust storms for large geographic areas poses several critical challenges:

- Simulating dust storm phenomena is complex and computing intensive (Xie et al. 2010). The periodic phenomena simulation requires repeating the same set of numerical calculations with a time interval for many times. Figure 10.1 shows a typical procedure for a Nonhydrostatic Mesoscale Model (NMM)-dust model (Huang et al. 2013a), which is a popular dust storm forecasting model. Each iteration includes 21 module calculations. For a given domain size, the computing complexity of an atmospheric model behaves as a function of where n is the grid dimension, including one time dimension, two horizontal dimensions, and one vertical dimension (Baillie et al. 1995). If the time dimension is extended, more iterations are needed. Doubling the geographic area on the horizontal direction would result in a fourfold increase in computing cost. Doubling the spatial resolution (e.g., from 10 km to 5 km) would result in an eightfold increase in computational cost because it would also require doubling the time step to keep the model accuracy (Baillie et al. 1995). High performance computing is used to perform dust storm simulation for large geographical areas with high resolutions (Huang et al. 2013a). NMM-dust (Huang et al. 2013a) is parallelized to leverage high performance computing (HPC) by decomposing the domain into multiple subdomains and distributing the computing load of each subdomain onto one CPU core as a process. The core that processes a subdomain must communicate with the neighboring cores for synchronization. Subroutines shown in Figure 10.1 in gray boxes require intensive data exchange and synchronization if parallelized (Huang et al. 2013a).

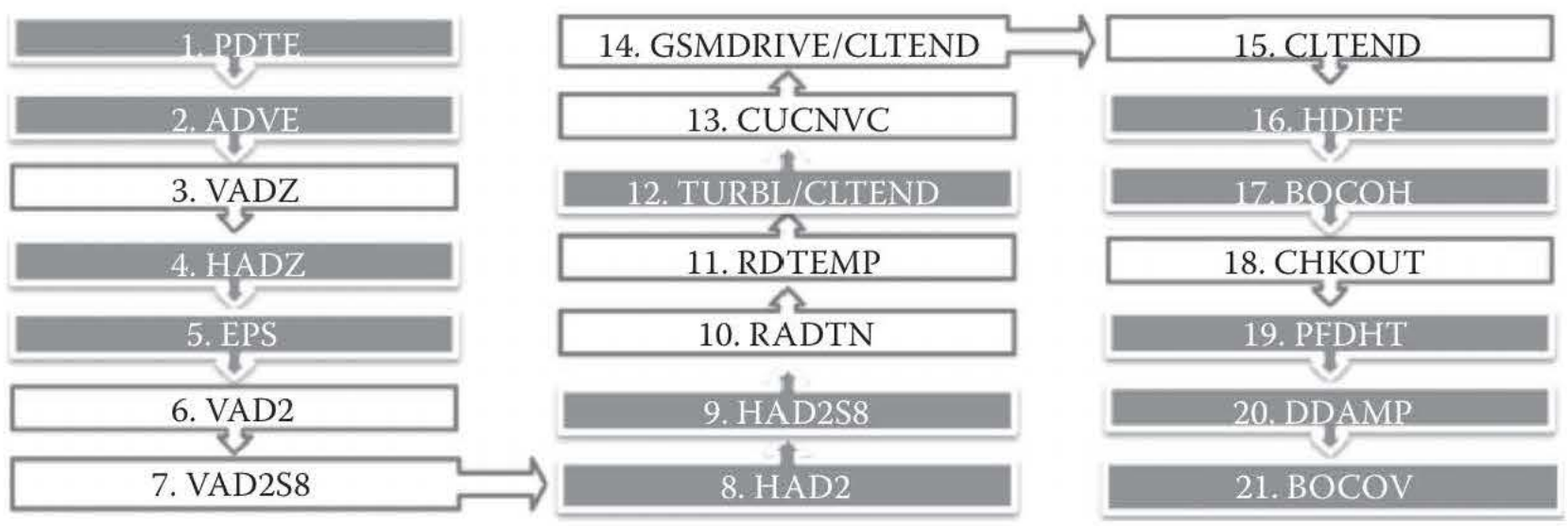

Figure 10.1 Computing subroutines for the NMM-dust model. (The subroutines in gray boxes require communication and synchronization.) (From Huang et al. 2013a.)

- Dust storm forecasting is a time critical task that needs to be finished in a limited time frame. For example, a two-hour computing limit is recommended for one-day forecasting to make the results useful (Drake and Foster 1995). Limited geographic area and/or resolution forecasting is usually performed to complete the simulations within the time limit (Wolters, Cats, and Gustafsson 1995). However, a zip code level resolution is needed for dust storm forecasting to support emergency decision making for governmental agencies, such as preparing medications for public health (Xie et al. 2010). Figure 10.2 shows the computing time required for a 24-hour forecasting over different domain sizes on the horizontal directions and with the same vertical layers (37 layers) and spatial resolution (3 km). More than 12.6 hours are needed to forecast a 10 × 10 degree domain size using an HPC cluster with 25 computing nodes (Yang et al. 2011). It is estimated that forecasting the whole southwest United States with a domain size of 37 × 20 degrees would require about 93 hours. Such computing performance is not acceptable because we would be forecasting the dust event in the past.
- Dust storms are disruptive events. It is estimated that the total time of dust storms in one year was generally less than 90 hours and only takes less than 1% of one year assuming that each dust storm lasts on average two hours (NOAA 2011). Therefore, a forecasting system for such events would expect different computing and access requirements during different times of the year and even different hours within a day.

To tackle these challenges, cloud computing can be leveraged. Cloud services can handle the spike and disruptive computing requirements with elastic and on-demand computing resources (Huang et al. 2013b). Additionally, with the development of cloud computing technologies such as virtualization and the general advancement of cloud infrastructure, cloud services are ready to support computing intensive applications.

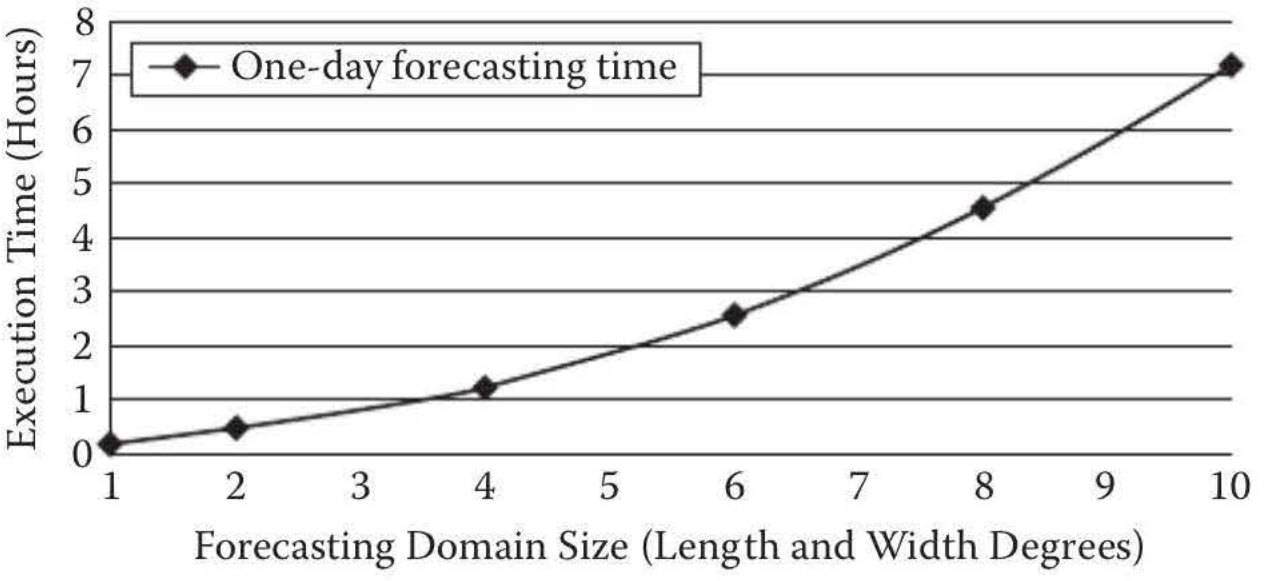

Figure 10.2 Execution time for different geographic domain forecasts. (From Huang et al. 2013b.)

For example, Amazon EC2 offers cluster instances with a 10 Gbps network connection. Each cluster instance has a minimum of 23 GB RAM, and two quad-core processors with a clock speed of 2.93 GHz. Such hardware and network configurations are far better than grid computing environments with heterogeneous computing resources and a Wide Area Network (WAN) connection, and even better than most private homogenous HPC cluster configurations. This makes cloud computing a new advantageous computing paradigm to resolve scientific problems that are computing intensive and traditionally require a special high-performance cluster (Rehr et al. 2010).

10.2 DEPLOYMENT AND OPTIMIZATION

10.2.1 General workflow

The general steps to deploy a dust storm model onto Amazon EC2 are as follows (Figure 10.3):

Step 1. Authorize network access—Network access authorization was introduced in Chapter 4. Port 22 is opened to enable communication between the local server and EC2 instances through Secure Shell (SSH).

Step 2. Launch an instance—Launch an instance as the head node of the dust storm simulation task (see Chapter 4, Section 4.3 for reference). A cluster instance with eight CPU cores and 23 GB of memory is chosen.

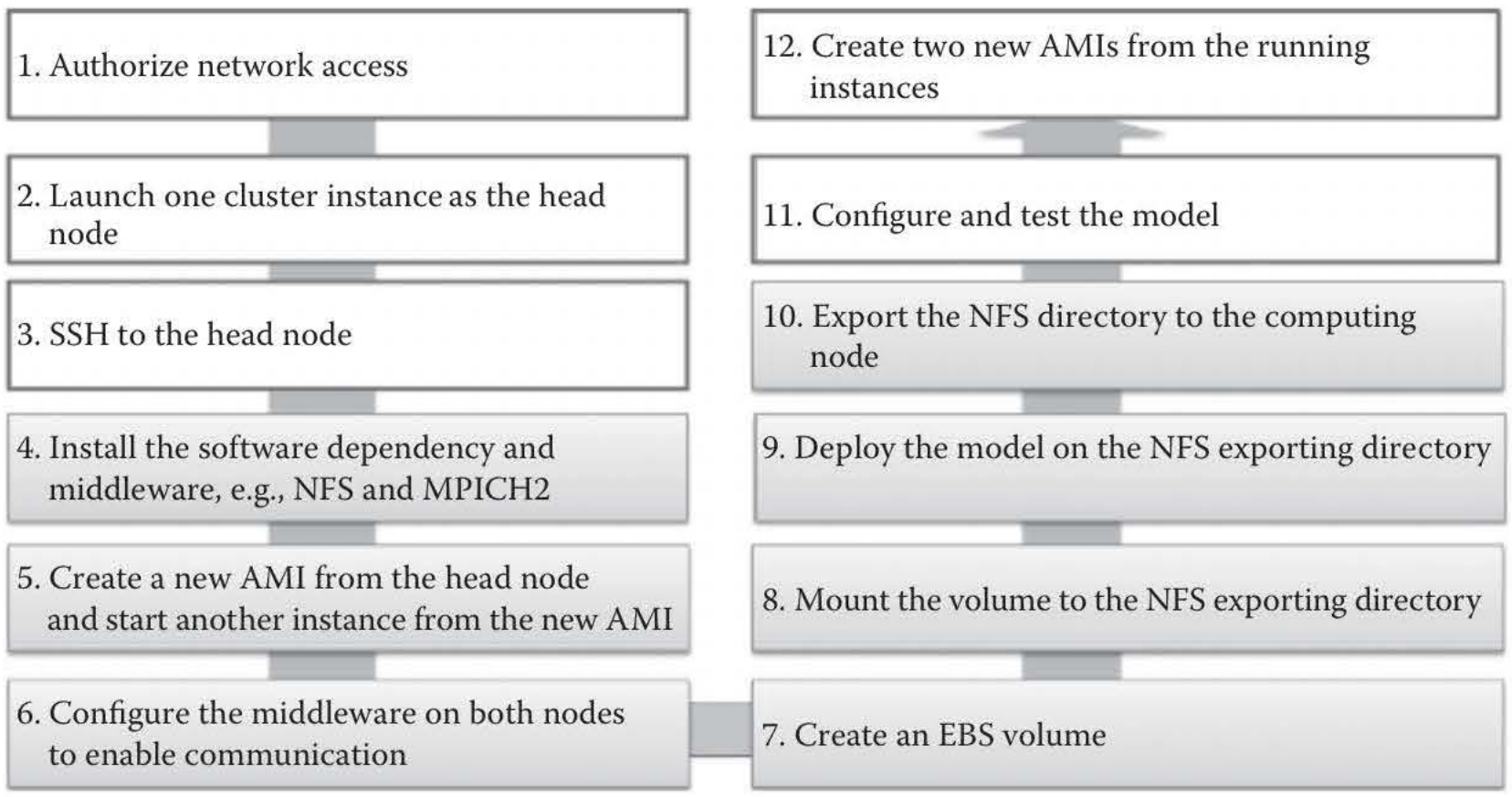

Figure 10.3 The process of deploying the dust storm model onto Amazon EC2. (Gray boxes indicate that the step requires special consideration.)

Step 3. SSH to the instance. Use an SSH tool to access the instance created in Step 2. Instance access was discussed in detail in Chapter 4, Section 4.3.

Step 4. Install software dependencies—Dust storm model execution requires that the Network File System (NFS) and MPICH2[1] are pre-installed on the head node and computing nodes. NFS enables all other computing instances to share the software package and model data. By using NFS, we only need to set up the model on the head node while other computing instances can share this environment without installing and configuring a model run environment. MPICH2 is one of the most popular implementations of Message Passing Interface (MPI). The following command line introduces the process of installing NFS and MPICH2 in an Amazon EC2 instance with CentOS 5.6 as the operation system.

```
[root@domU-head~] yum install gcc gcc-c++ autoconf
  automake
[root@domU-head~] yum -y install nfs-utilsnfs-utils-
  lib sytem-config-nfs #install NFS
[root@domU-head~] tar -zvxf mpich2-1.5.tar.gz
  download MPICH2 package and unzip
[root@domU-head~] mkdir /home/clouduser/mpich2-
  install # create an installation directory
[root@domU-head~] cd mpich2-1.5.tar.gz
[root@domU-head~] ./configure -prefix=/home/
  clouduser/mpich2-install
[root@domU-head~] make # Build MPICH2
[root@domU-head~] make install # Install MPICH2
```

After NFS is installed, the following commands can be used to configure and start NFS.

```
[root@domU-head ~] mkidr /headMnt # create a NFS
     export directory
[root@domU-head ~] echo "/headMnt *(rw) " >> /etc/
     exports
[root@domU-head ~] exportfs -ra
[root@domU-head ~] service nfs start #start up NFS
```

Step 5. Create an AMI and a computing node—Build an AMI based on the configured instance in Step 4. Use this new AMI to launch one instance as the computing node.

Step 6. Configure the head node and computing node—This step establishes network communication between the head node and computing nodes. First, use SSH to access the head node

[1] See MPICH2 at http://www.mcs.anl.gov/research/projects/mpich2staging/goodell/index.php.

and add the IP of the computing node to the host list. Then create a public key at the head node. Finally, use SSH to access each computing node, and copy the head node's public key to the authorized key list.

```
[root@domU-headMnt ~] vi /etc/hosts #access to the
  hosts list of the head node
[root@domU-headMnt ~] ssh-keygen -t rsa #create a
  public key at the head node

[root@domU-computing~] mkdir -p /root/.ssh/
[root@domU-computing ~] scp root@domU-head: /root/.
  ssh/id_ras.pub /root/.ssh/ #copy the public key
  from the head node to the computing node
  [root@domU-computing ~] cat /root/.ssh/id_ras.pub >>
    /root/.ssh/authorized_keys
```

Step 7. Create an EBS volume and attach it to the head node—Please refer to Chapter 8, Section 8.3 for information regarding creating a new EBS volume and attaching it to a running instance.

Step 8. Mount the volume to the NFS export directory—Please refer to Chapter 8, Section 8.3 for mounting the volume to an instance.

Step 9. Deploy the model on the NFS export directory—Transfer the dust storm model (refer to Chapter 5, Section 5.3.1 for large data transfer between the local server and EC2 instances) to the NFS export directory of the head node, which is `/headMnt` in this case.

Step 10. Export the NFS directory to the computing node—This step establishes the NFS connection between the head node and computing node, which enables the computing node to share the package suites from the head node. Use SSH to access the computing node, and mount the volume to the NFS directory. The command below shows an example of mounting a computing node volume to the NFS directory. `domU-head` is the domain name of the head node. `/headMnt` is the file path of the NFS directory in the head and computing nodes. After issuing the command to mount the NFS directory from the head node to the computing node, cloud consumers can go to the `/headMnt` directory and check if the model is available on the computing node.

```
[root@domU-computing ~] mkdir //headMnt # create the
  directory
[root@domU-computing ~] mount -t nfs -o rw domU-head:/
  headMnt //headMnt #Mount the volume to the NFS export
  directory
```

Step 11. Configure and test the model—Create a host file with a list of either IPs or domain names that can run MPI tasks (e.g., `mpd.hosts`), and add the private IP of each computing node to this file as shown below (including two computing nodes).

```
199.26.254.161 # node1
199.26.254.162 # node2
```

On the head node, start the MPI connection. The below command shows how to start the MPICH2 service. The parameter `-n 3` is the number of computing nodes plus head node, and `mpd.hosts` is the machine configuration file. The below command also shows an example of executing the model script: `run_test.sh` is the model execution script, and `out.log` is the log file to store model execution information.

```
[root@domU-head~] mpdboot -n 3 -f mpd.hosts –v #
     start up MPICH2
[root@domU-xxx~] ./run_test.sh >& out.log & # Run the
     model
```

Step 12. Create an AMI from the running instance—Finally, two new AMIs can be created based on the running head node and computing node separately (refer to Chapter 4, Section 4.3 for how to create an AMI).

10.2.2 Special considerations

In addition to the general deployment steps, several configurations are unique to dust storm models with high computing power requirements.

- *Configure a virtual cluster environment*—To support dust storm forecasting with cloud services, a virtual cluster environment should be set up. As introduced in Chapter 5, Section 5.3, one or more virtual instances should be used with middleware installed and configured to construct a virtual cluster. In this chapter, MPICH2 is used as the middleware (Chapter 5 uses Condor) to dispatch the model run tasks and collect the results. In order to save the time to configure the computing nodes, an image from the head node at Step 5 (Figure 10.3) can be built to launch a computing node. In addition, the communication overhead can be greatly reduced if all instances are launched within the same data centers and on physically close machines. Therefore, cloud consumers should create a placement group and launch multiple cluster instances into the placement group (Figure 10.4). A cluster placement group is a logical grouping of cluster

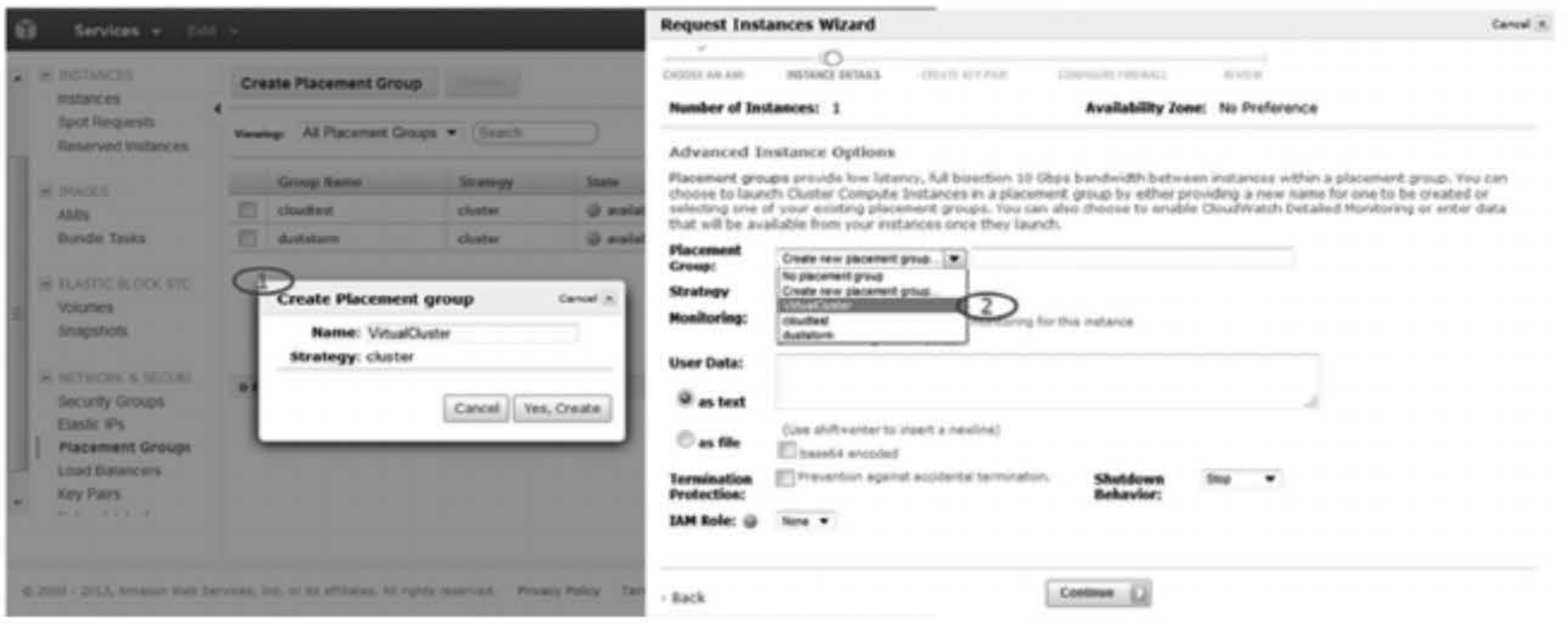

Figure 10.4 (See color insert.) Create a placement group and place the instances within the same group.

instances within a single EC2 data center. A placement group cannot span multiple data centers. In this way, consumers can logically group cluster instances into clusters in the same data center. It enables many HPC applications to get the full-bisection bandwidth and low-latency network performance required for tightly coupled, node-to-node communication. Amazon recommends that the cloud consumers launch the minimum number of instances that are required for supporting the applications in the placement group in a single launch request. If consumers launch only a few instances and try to add more instances to the placement group later, they may get an "Insufficient Capacity" error because no physical resources are available to create virtual machines on the same HPC rack. If consumers do receive an "Insufficient Capacity" error, stop and restart the instances in the placement group, and try to add more instances again.[1]

Cloud consumers can also use the following command line to create a placement group (i.e., VirtualCluster) and place all instances (e.g., two instances in this case in the same group).

```
PROMPT>ec2-create-placement-group VirtualCluster -s
cluster
PROMPT>ec2-run-instances ami-xxx -n 2 --placement-
group VirtualCluster
```

- *Loosely coupled nested modeling and cloud computing*—During the loosely coupled nested model test (Huang et al. 2013a), a large domain is reduced into multiple small regions, which are identified by the coarse model ETA-8bin. A large amount of computing resources should be

[1] See AWS Elastic Cloud Compute at http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using_cluster_computing.html.

leveraged to concurrently run these 18 areas of interest (AOIs), each requiring high-resolution forecasting. Elastic Amazon EC2 cluster instances are utilized to enable the nested modeling approach to its full extent. This is because multiple virtual clusters can be constructed in the Amazon EC2 platform with each virtual cluster running one subregion directly (Section 10.3.3 covers the performance details).

- *Auto-scaling*—Since this is not a Web application, the method to trigger auto-scaling resources through CloudFormation (Chapter 8, Section 8.3) is not suitable. The cloud consumer would have to write scripts with the EC2 APIs to launch more instances to handle the computing when needed.

10.2.3 Summary of the differences from the general steps in Chapter 5

Compared to the general steps discussed in Chapter 5, the entire deployment procedure of a dust storm model to a cloud service has the following characteristics:

- *Handling computing intensity*—This dust storm model requires leveraging multiple virtual instances to run the simulation concurrently. Therefore, a virtual cluster environment should be set up with the middleware MPICH2 installed and configured properly on the head node and computing nodes.
- *Handling communication intensity*—A placement group (Figure 10.4) should be created and all instances for the virtual cluster should be within the same group to reduce the communication overhead.

10.3 DEMONSTRATION

10.3.1 Phoenix dust storm event

The State of Arizona is home to some of the most spectacular displays of blowing dust in the United States. Dangerous dust storms impact the state several times per year and cause numerous economic losses (Shoemaker and Davis 2008). The largest of these dust storms is called a *haboob*, which is the most common in the central deserts of Arizona during its summer season, with the frequency of occurrence peaking in late July and early August. The city of Phoenix, Arizona, experiences on average about three haboobs per year during the months of June to September (Shoemaker and Davis 2008). Figure 10.5 shows a classic haboob that moved through the Phoenix area on July 1, 2007. This chapter uses this event as an example to show how a dust storm model can simulate a dust storm event and how cloud computing can help perform such a simulation.

Figure 10.5 (See color insert.) Photograph of a haboob that hit Phoenix, Arizona, on July 1, 2007. (Courtesy of Osha Gray Davidson, *Mother Jones*, San Francisco, CA at http://www.motherjones.com/blue-marble/2009/09/australias-climate-chaos.)

10.3.2 Simulation result

Figure 10.6 shows a time-frame result at 3 A.M., July 2, 2007, produced by ETA-8bin (a low-resolution model) (Huang et al. 2013a) and NMM-dust models in different spatial resolutions. The model results show similar patterns for the dust storm's area and NMM-dust results with a 3-km resolution, which picked up much more detailed information about the dust concentrations.

10.3.3 Performance

Elastic Amazon EC2 cluster instances are utilized to enable the loosely coupled nesting approach to its full extent (Huang et al. 2013b). During the loosely coupled nesting approach, ETA-8bin performs a quick forecasting with low resolution (22 km) to identify potential hotspots with high dust concentration. NMM-dust will perform high resolution (3 km) forecasting over the much smaller hotspots in parallel to reduce computational requirements and computing time.

In this case, 18 AOIs are identified by the coarse model ETA-8bin. Figure 10.7 shows the width and length distribution of those 18 AOIs. It is observed that most of them are distributed within a 2 × 2 degree scope. Therefore, 18 cluster instances are launched from the AMI configured through the steps in Section 10.2 and each instance is responsible for handling the simulation for one AOI.

Figure 10.8 shows the execution time for each instance by 18 Amazon instances, with each instance simulating one AOI region for a 24-hour forecast. The results reveal that most of the AOIs can be successfully

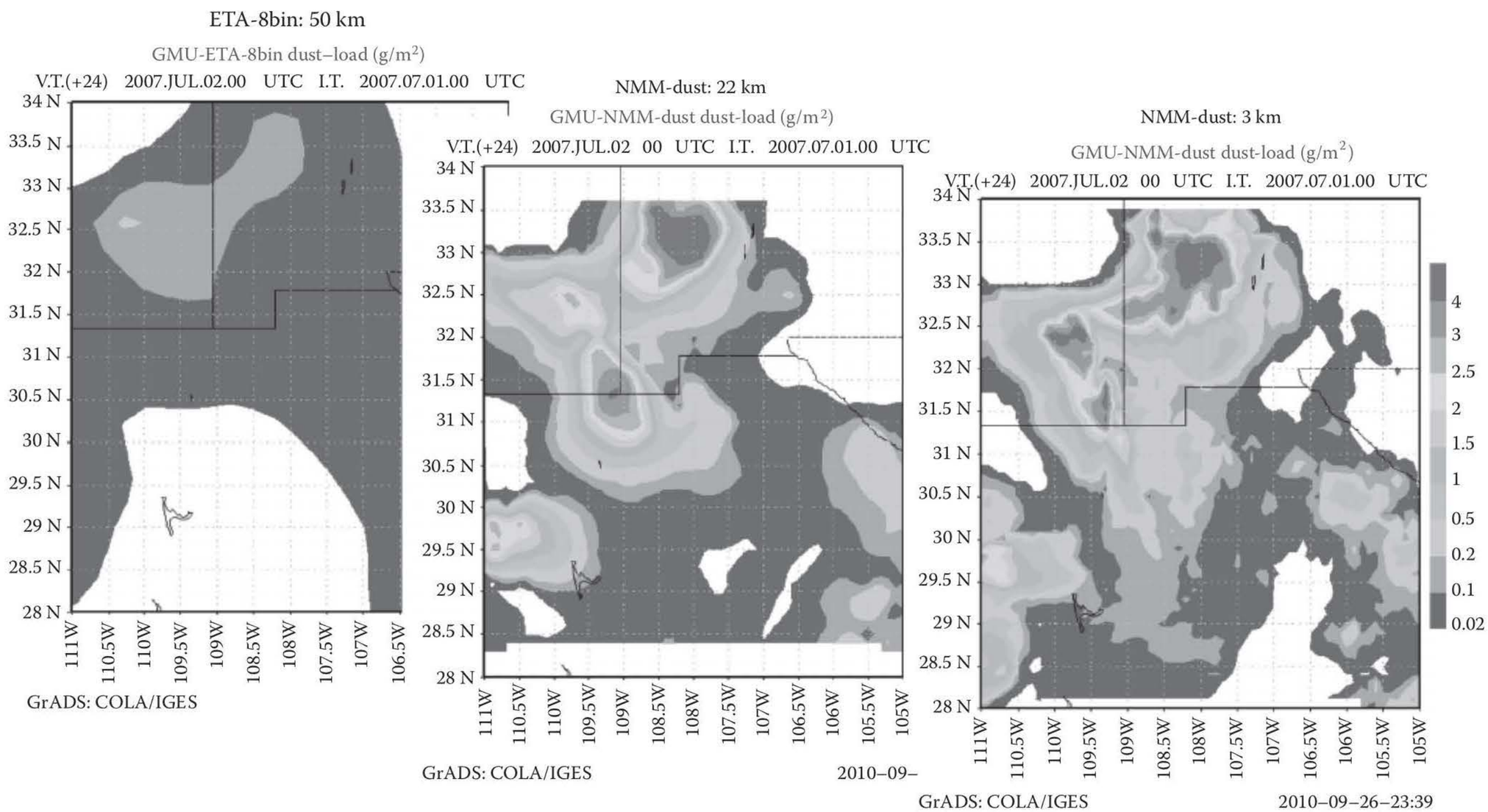

Figure 10.6 (See color insert.) Comparison of the simulation results by ETA-8bin and NMM-dust AOI 10, 11, 12, and 13 at 3 a.m., July 2, 2007.

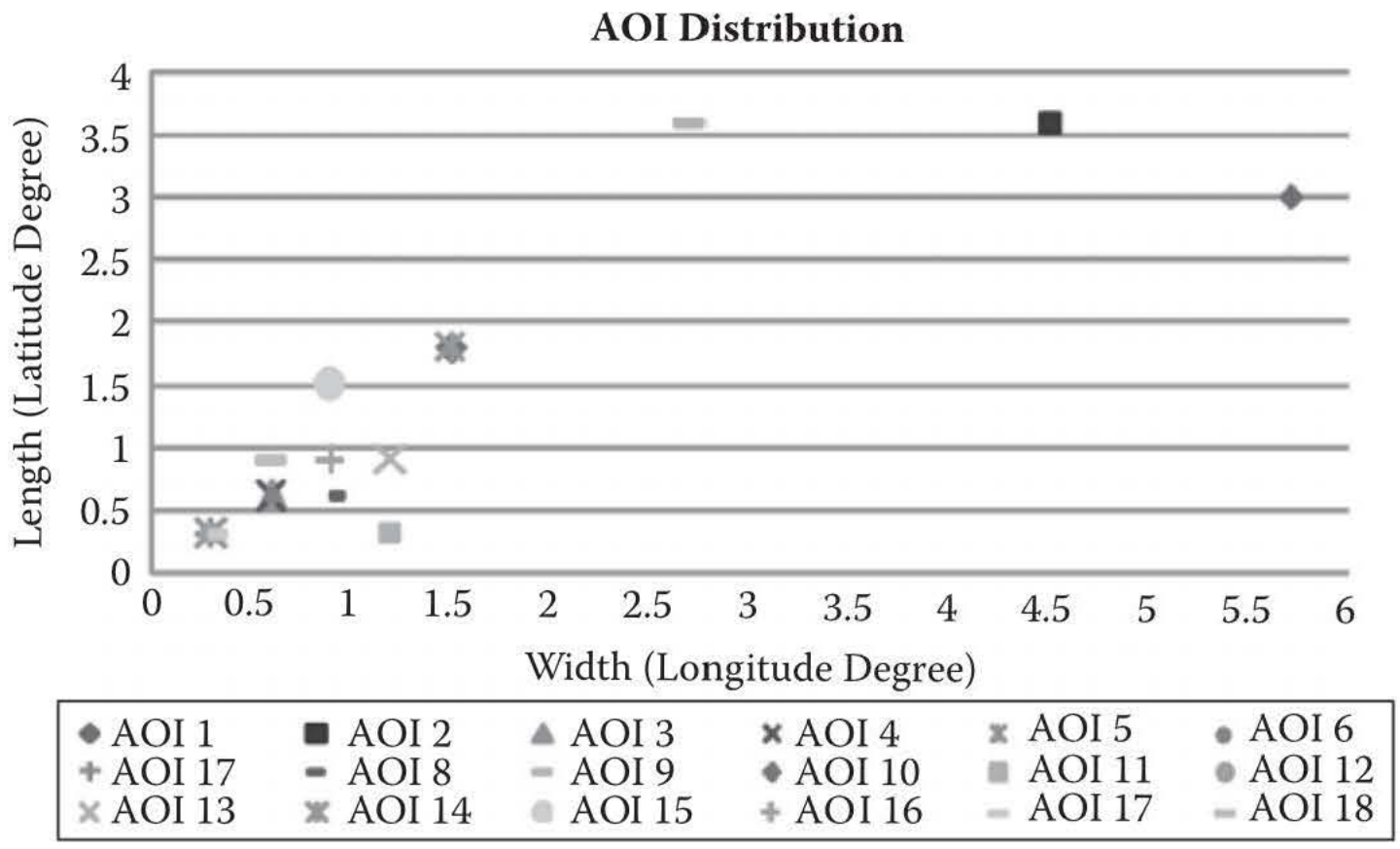

Figure 10.7 (See color insert.) AOI distribution identified by the ETA-8bin.

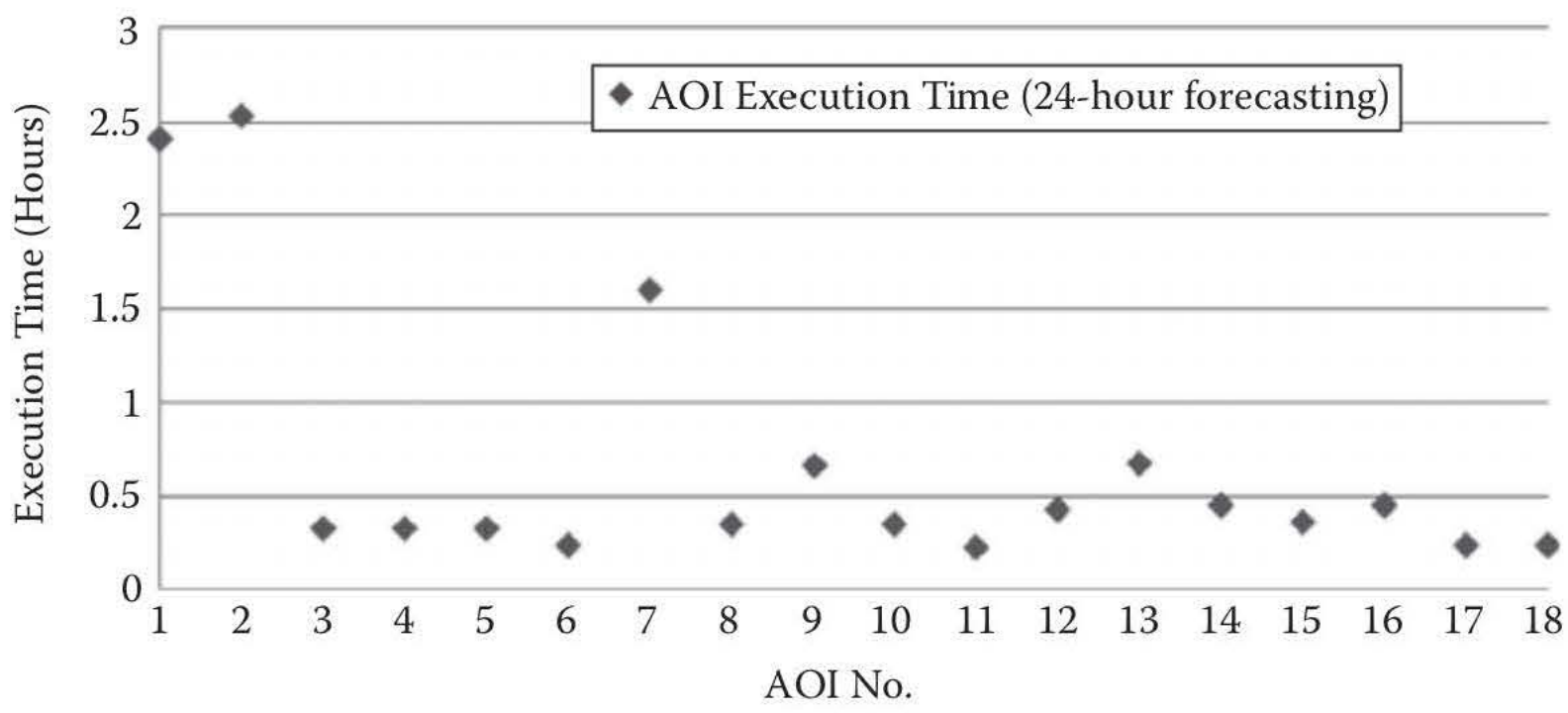

Figure 10.8 Eighteen Amazon cluster instances are launched to run 18 AOIs in parallel with each instance simulating one AOI region for a 24-hour forecast.

completed within one hour for a 24-hour forecasting. However, two AOIs cannot be successfully completed within two hours. Therefore, more computing resources and optimizations should be integrated to enable the computing to achieve the two hours for 24-hour simulation time constraint.

Two and three cluster instances are used to further test the computability of those two subregions. Figure 10.9 shows the performance of Amazon EC2 cloud services when utilizing two and three instances for each virtual cluster with and without optimizations for three-hour forecasting over the first subregion. It is observed that the system cannot achieve the 0.25-hour time constraints even with three cluster instances involved. Therefore, the two instances should be optimized through better parallelization and

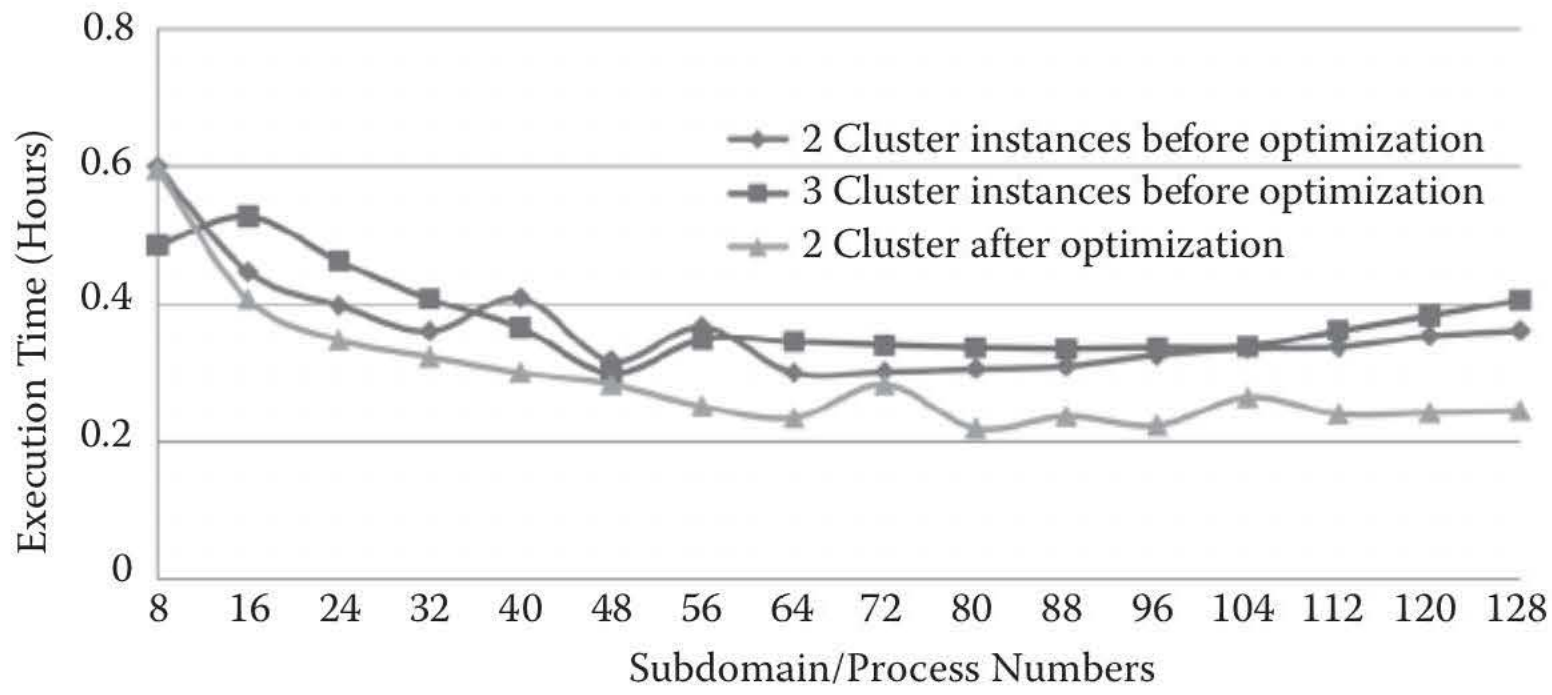

Figure 10.9 Performance comparisons of Amazon cluster instances before and after optimizations.

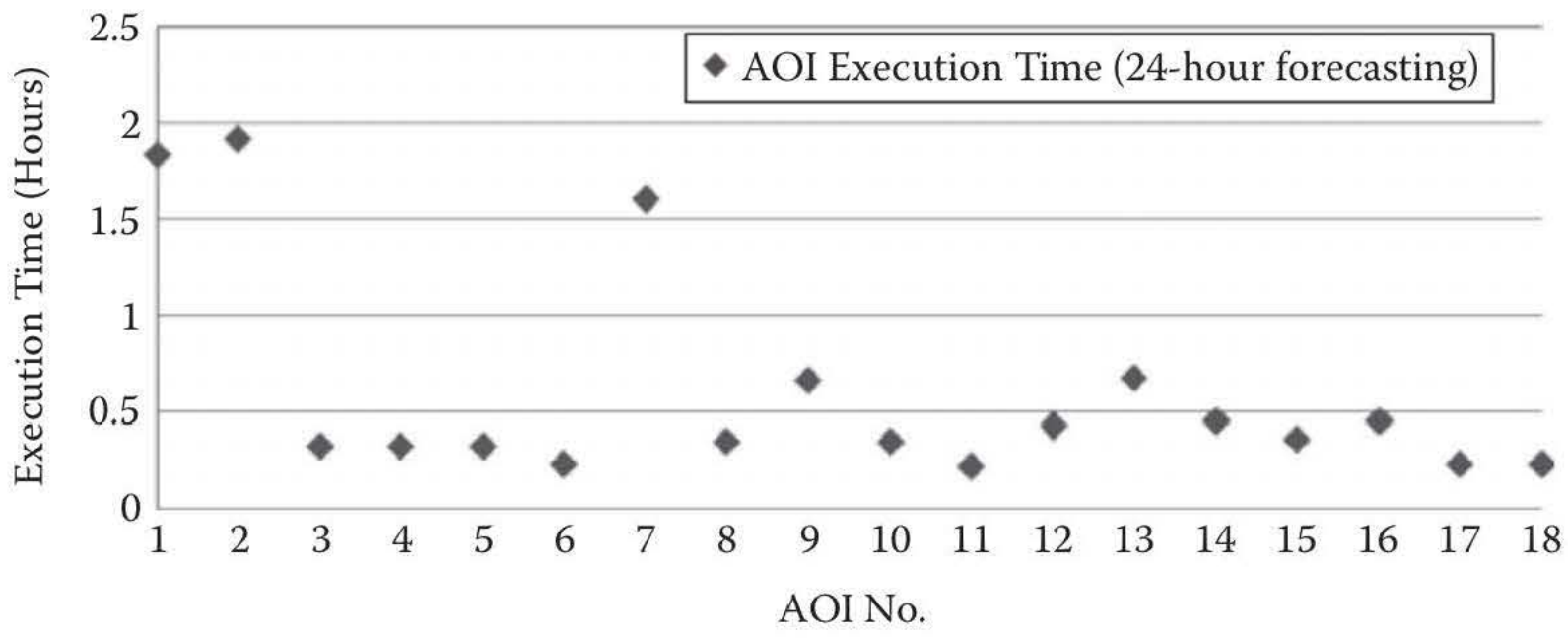

Figure 10.10 Eighteen subregions run on Amazon EC2 cloud service after optimization. Both AOI 1 and 2 utilize two optimized cluster instances.

scheduling strategies, including neighbor mapping and spatiotemporal optimization strategies (Yang et al. 2011). Figure 10.9 shows that the two optimized instances can successfully complete the forecasting within 0.25 hours if using more than 64 processes. Finally, with these two instances starting 64 processes, it is observed that the two AOIs can be successfully completed within two hours (Figure 10.10).

10.3.4 Cost-efficiency

For an operational dust storm forecasting system, usually an EC2 instance is reserved for long-term forecasting with a much lower price compared to an on-demand instance. It costs only $0.16 per hour for general public user access and low-resolution forecasting. When the dust storm events occur, a large group of instances would be started to respond to concurrent user

access and high-resolution forecasting. It is observed that the hourly cost of this cluster is much higher than that of EC2 cloud service when there is no dust storm event occurring and only one reserved instance is needed (Huang et al. 2013b). Under this situation, the cluster's hourly costs are more than 896 times than that of the single reserved instance.

Typically, there are around 45 dust storm events occurring in the United States in total per year (NOAA 2011). The yearly cost of a local cluster is around 12.7 times higher than that of the EC2 cloud service if 28 EC2 instances (with 400 CPU cores) are leveraged to handle the high resolution and concurrent computing requirements for a duration of 48 hours. With much lower costs, the cloud service can provision the same computing power as a local cluster within a shorter time period in responding to the dust storm events. These cost comparison results indicate that it is economical to use cloud computing by maintaining low access and resolution forecasting while invoking a large scale of computing power to perform high-resolution forecasting for large geographic areas when needed.

10.4 CONCLUSION

Dust storm events exhibit an interannual variability and are typical disruptive events. The computing pool to host an operational dust storm forecasting system should also support the disruptiveness by (1) scaling up to respond to potential high resolution and concurrent computing requests when dust storm events happen, and (2) scaling down when no dust storm events occur to save costs and energy. Cloud computing is used to support dust storm forecasting by:

- Being capable of provisioning a large amount of computing power in a few minutes to satisfy the disruptive computing requirements of dust storm forecasting (Section 10.3.2).
- Economically sustaining low access rates and low-resolution models, while still being able to invoke a large amount of computing power to perform high-resolution forecasting for large public access when needed (Section 10.3.3).

More advanced configurations such as a better file system design to reduce I/O overhead (Huang et al. 2013a) can better support large-scale simulations such as dust storm forecasting. In the cluster computing architecture, each computing node is usually designed to access the same remote data storage to execute tasks in parallel. In this example, NFS is used to share the storage and to ensure the synchronization of data access. However, different file systems could have great impacts on the performance of I/O and data intensive applications. Therefore, cloud consumers can explore other solutions such as the Parallel Virtual File System version 2 (PVFS2) (Latham et al. 2010).

10.5 SUMMARY

This chapter discusses how cloud services are capable of supporting the forecasting of disruptive events such as dust storms, using Amazon EC2 as an example. Section 10.1 explains the background and motivation of conducting the dust storm forecasting, and presents the major challenges in performing dust storm forecasting. The details of the configuration and deployment of a dust storm model onto the EC2 cloud service are described in Section 10.2. Section 10.3 demonstrates how a cloud service can enable dust storm forecasting using the December 22, 2009 Phoenix dust storm as an example. Section 10.4 concludes the improvements by using cloud computing to support such large-scale simulation and forecasting as well as future research directions.

10.6 PROBLEMS

1. What are the computing challenges for dust storm forecasting?
2. What are the general steps to deploy dust storm simulation onto the cloud?
3. Which instance type is better for dust storm forecasting; regular instance or HPC instance? Why?
4. How to configure a virtual high performance computing (HPC) cluster to support computing-intensive applications?
5. How is Elastic Block Storage (EBS) service used in supporting the dust storm model deployment to the cloud?
6. How do you create a placement group for HPC instances using both Amazon Web Console management and command line tools? Why do we need this step?
7. Compared to the deployment workflow for general applications introduced in Chapter 5, what are the special considerations for a dust storm model?
8. Why can cloud computing achieve cost-efficiency?
9. Why does cloud computing provide a good solution to support a disruptive event (e.g., dust storm) simulation?

REFERENCES

Baillie, C. F., A. E. MacDonald, and S. Sun, 1995. QNH: A portable, massively parallel multi-scale meteorological model. *Proceedings of the 4th Int'l Conference on the Applications of High Performance Computers in Engineering*, June 19–21. Milan, Italy.

Goudie, A. S. and N. J. Middleton. 1992. The changing frequency of dust storms through time. *Climatic Change* 20, no. 3: 197–225.

Drake, J. and I. Foster. 1995. Introduction to the special issue on parallel computing in climate and weather modeling. *Parallel Computing* 21, no. 10: 1539–1544.

Huang, Q., C. Yang, K. Benedict, A. Rezgui, J. Xie, J. Xia, and S. Chen. 2013a. Using adaptively coupled models and high-performance computing for enabling the computability of dust storm forecasting. *International Journal of Geographic Information Science* 27, no. 4: 765–784.

Huang, Q., C. Yang, K. Benedict, S. Chen, A. Rezgui, and J. Xie. 2013b. Enabling dust storm forecasting using cloud computing. *International Journal of Digital Earth* 6, 4: 338–355.

Latham, R. et al. 2010. A next-generation parallel file system for Linux cluster. *Linux World Magazine* 2, no. 1: 54–57.

NOAA. 2011. Dust Storm Database [online]. http://www4.ncdc.noaa. gov/cgi-win/wwcgi.dll?wwevent_storms (accessed October 30, 2011 and 2013).

Rehr, J. J., F. D. Vila, J. P. Gardner, L. Svec, and M. Prange. 2010. Scientific computing in the cloud. *Computing in Science & Engineering* 12, no. 3: 34–43.

Shoemaker, C. and J. Davis. 2008. Hazardous Weather Climatology for Arizona, NOAA Technical Memorandum. 2008. http://www.wrh.noaa.gov/wrh/tech-Memos/TM-282.pdf (accessed March 10, 2013).

WMO (World Meterological Organization). 2012. WMO Sand and Dust Storm Warning Advisory and Assessment System (SDS-WAS): Science and Implementation Plan 2011–2015. http://www.wmo.int/pages/prog/arep/ wwrp/new/documents/SDS_WAS_implementation_plan_01052012.pdf (accessed March 30, 2013).

Wolters, L., G. Cats, and N. Gustafsson. 1995. Data-parallel numerical weather forecasting. *Science Program* 4, no. 3: 141–153.

Xie, J., C. Yang, B. Zhou, and Q. Huang. 2010. High-performance computing for the simulation of dust storms. *Computers, Environment and Urban Systems* 34, no. 4: 278–290.

Yang, C., H. Wu, Q. Huang, Z. Li, and J. Li. 2011. Using spatial principles to optimize distributed computing for enabling the physical science discoveries. *Proceedings of the National Academy of Sciences* 108, no. 14: 5498–5503.