

# HiCloud: Taming Clouds on the Clouds

Han Zhao, Xin Yang and Xiaolin Li  
S3Lab, CS Dept.  
Oklahoma State University  
Stillwater, OK 74078, USA  
{haz, xiny, xiaolin}@cs.okstate.edu

H. Howie Huang  
ECE Dept.  
George Washington University  
Washington DC 20052, USA  
howie@gwu.edu

Ming Xue  
School of Meteorology and CAPS  
University of Oklahoma  
Norman OK 73072, USA  
mxue@ou.edu

## I. INTRODUCTION

The proposed project is aimed to extend current computationally-intensive weather forecasting research on geographically distributed resources for faster results delivery and improved scalability. To achieve this goal we take advantage of the emerging cloud computing paradigm [1], [2], [3], [4], [5] to enable very large-scale weather forecasting applications in the proposed cloud infrastructure. Such an integration is promising but challenging due to increasing complexity and overheads in resource management and job scheduling. This project involves interdisciplinary and multi-site collaboration. The main tasks are as follows.

- Deploy computational platforms at multiple geographically distributed sites as Cloud Service Accessing Points (CSAP).
- Investigate scheduling methods for decomposing computational jobs and dispatching.
- Develop extensible user interfaces and runtime middleware on the basis of developed open source cloud computing infrastructures.

A large number of applications require computation of highly calculation-intensive tasks and cooperation from multiple disciplines and locations. Example of such applications including quantum mechanical physics, molecular modeling, nuclear fusion simulation, cryptanalysis and weather forecasting. In this proposed project we focus on storm-scale ensemble and high-resolution forecasts in which real-time visualization of computational results are vital for postprocessing analysis [6], [7]. In convective-scale hazardous weather forecasting, the orders-of-magnitude of uncertainty and inaccuracy is too high to be resolved using ordinary computing server. In addition, the high-nonlinearity of the weather systems at large scale makes it even harder for decision making. As a result, typical experiments conduct on high performance supercomputers consisting of thousands CPU cores and fast internal connection. For example, based on report of 2008 spring experiment on NOAA hazardous weather testbed [6], the Weather Research and Forecast (WRF) model runs on the Pittsburgh Supercomputing Center (PSC). A total number of about 1500 CPUs of a Cray XT-3 supercomputing were reserved for up to 8 hours each night for the forecasts, and the data archived were about 1 terabytes a day. The computing process is labor-intensive and time consuming. Hence, we

propose to extend current forecasting applications to larger scale through cloud computing and explore the feasibility of scheduling and coordination in new organization of cloud resources.

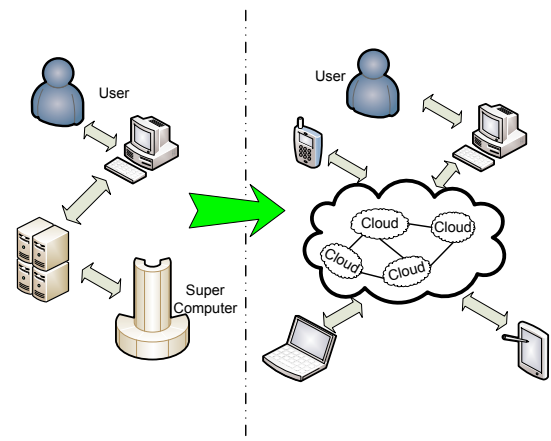


Fig. 1. Migrate HRWF applications from supercomputers to clouds

Cloud computing has become a new paradigm in both academia and industry in recent two years, delivering services from dynamic scalable and virtualized computing and storage resources over the Internet [8], [9]. Unlike conventional application-oriented supercomputing paradigms and grid computing [10], [11], [12], cloud computing built upon Web 2.0 is service oriented. Generally it offers three different levels of services [13]: Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS). The later two service types are more restricted in providing platforms and special purpose software for remote access and application development. For example, Google App Engine [14] supports users to develop and maintain Web applications on resources provided by Google without additional downloading. The first type of service IaaS is at the lowest level and enables more flexible in delivering dynamically scaled infrastructure. In IaaS all equipments are packed as services and offered to users in a resource usage-based pricing model. Examples of IaaS are Amazon Elastic Compute Cloud (EC2) [15] and Eucalyptus [16]. Amazon EC2 allows users to create Amazon Machine Image (AMI) containing user configuration setting and access computing resources using simple Web service interfaces. Eucalyptus is

open source and is developed for academic purpose, allowing researchers to build their own cloud infrastructure through compatible Amazon EC2 interfaces. In this project we will utilize Eucalyptus to build a cloud infrastructure for weather forecasting applications. Figure 1 demonstrates our proposed solution for migrating High Resolution Weather Forecasting (HRWF) applications to cloud infrastructure. To address the scalability issues we plan to deploy the cloud infrastructure in multiple participating sites, including OSU, OU and.... We also plan to implement Web-based user interfaces to aid remote user access.

The rest of this proposal is organized as follows. Section II presents motivating applications and research challenges. Section III presents the proposed solution. Section IV presents our research plan to fulfill the proposed research. Section V concludes the proposal.

## II. MOTIVATING APPLICATIONS AND RESEARCH CHALLENGES

### A. Motivating Applications

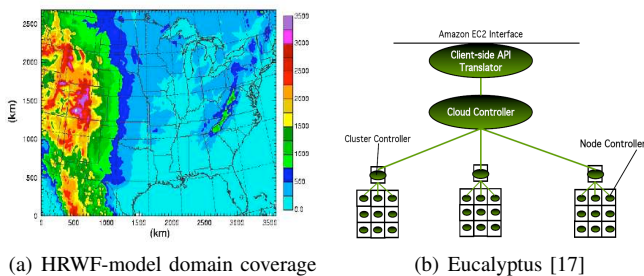


Fig. 2. Motivating applications

Two applications inspire us to develop large scale cloud-aware weather forecasting applications. Figure 2 demonstrates these two applications. In figure 2(a) a model domain coverage for 2008 season is generated by the forecast modeling application developed at University of Oklahoma. For daily storm-scale ensemble forecasts, NSF TeraGrid resources (Cray XT3) are reserved for model execution [7]. The execution typically will last for 6 to 10 hours and the final results will be put on public for further analysis when they become available. Eucalyptus [16] is an open source cloud computing architecture, which implements web services based cloud computing infrastructure. Other related open source project including Nimbus [18] and Enomalism [19]. The goal of Eucalyptus is to offer an experiment environment for Amazon EC2 and to foster research in utility and cloud computing. In Eucalyptus the architecture is hierarchical, as shown in Figure 2(b). Any request launched by user will interact with the Cloud Manager (CM) or Cloud Controller. CM will further contact Group Manager (GM) or Cluster Controller installed at front-end node of each cluster for resource reservation and task allocation. The user requests is fulfilled at each computing node, controlled by the Instance Manager (IM) or Node Controller. By providing virtual private network connection Eucalyptus provides easy

configuration and deployment of commercial like clouds in typical lab environments.

### B. Challenges

In this part we present key design issues as well as implementation challenges in HiCloud.

**Raw data slicing.** Unlike unstructured data produced by web applications (e.g. work count, reverse web link) and structured data (e.g. data warehouse), raw data of weather forecasting applications (WFRD) behaves randomly and somewhat regularly. Traditional database partitioning methods will not suit for WFRD. Thus we need to design and implement hybrid algorithms for WFRD slicing to cater to unique behaviors of HRWF outputs.

**Distributed file system.** The huge data produced by HRWF applications call for both well-managed and fault-tolerant distributed file storage. However, since WFRD is hybrid in nature, we argue that conventional distributed file system like Google's GFS [20] may restrict storing WFRD chunks in HiCloud. Recently Liu et al. [21] proposed DFS in GridBatch, which is an extension of GFS and is more competitive in organizing structured data, providing valuable results in setting up distributed storing for HiCloud.

**Client interface.** HiCloud is highly specialized, indicating that the web-based interface provided by current Eucalyptus release needs to be modified and customized for domain-specific requests.

**Security.** Security is urgent for HiCloud primarily due to two reasons: 1) unreliability of public network poses threats to requests submitted by computing launchers; 2) the use of both public network and virtual network calls for a unified security schema.

## III. PROPOSED SOLUTION

In this section, we present how to deploy current Weather Research and Forecasting applications onto existing cloud infrastructure. First, we give an overview of the proposed architecture by mapping key components onto multiple cloud layers. We then describe our plan of deployments and implementations.

### A. Conceptual Cloud Architecture

Figure 3 illustrates the conceptual architecture of the proposed cloud architecture for high resolution weather forecasting, called HiCloud. Our work mainly focuses on the layers connected by solid line. Motivated by the cloud computing stack [22], we map key components of weather forecasting applications onto this stack as follows (from top layer to bottom).

**Clients.** Cloud clients consist of both hardware and software and are dependent on service interfaces of cloud in task executing. Since we are using Eucalyptus as the underlying infrastructure, the environment is mostly academic-oriented representing typical research lab configuration where clusters are dominant. Web browsers installed on computing nodes and front-end will serve as software clients.

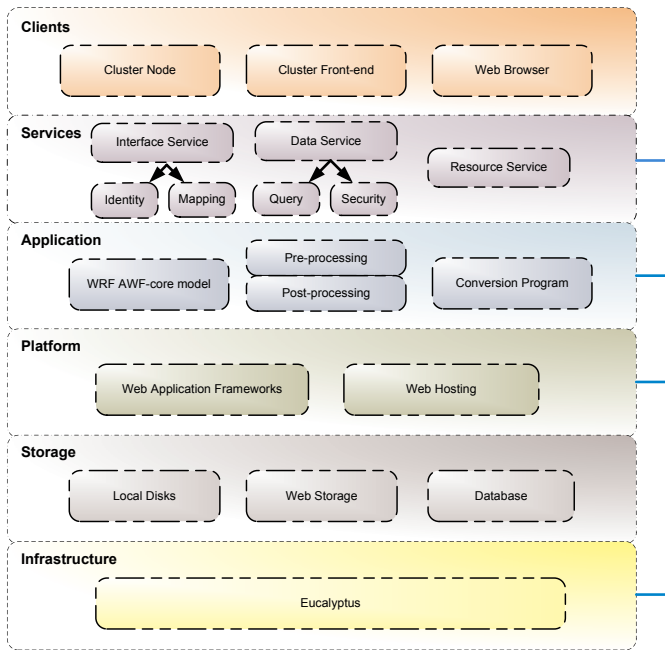


Fig. 3. Conceptual Architecture of HiCloud

**Services.** Eucalyptus Cloud Manager (CM) is in charge of service requests issued by user. Typically CM offers three-tiered services, namely interface service, data service and resource service. The services layer provides most of the flexibilities in Eucalyptus, allowing users to modify and tune existing implementation with multiple levels of granularity. The first tier interface service has three functions: 1) exposes visible interfaces to software clients; 2) manages authentication and internal protocol translation; 3) administrates user accounts. To implement HiCloud, it is vital to modify Eucalyptus implementation on interface services which utilize Amazon EC2 programmatic interfaces (SOAP and QUERY) and translate them into customized internal protocols. Besides we will integrate authentication identity and account administration tools into our implementation.

Data services manage user and system data and present data operations to users. With data services, resource abstraction is achieved, delivering uniform searching services. The security issues are manipulated as well in data service implementation. The resource services interact with lower physical resource by allocation and launching functions.

**Application.** Applications for the HiCloud project are forecasting softwares developed at Center for Analysis and Prediction of Storms (CAPS) at the University of Oklahoma, including modeling, pre and post processing and conversion programs used to run on supercomputers.

**Platform.** Platform for HiCloud includes web application frameworks and web hosting on which interfaces are developed and deployed. For example, Axis2 [23] is employed to provide core engine for web services. Jetty [24] is used as the web server for hosting. Most of the tools are open source projects which are easy to access and are easy to be tuned to specific design requirements.

**Storage.** Conventional weather forecasting application requires tera-byte level for data storage. In HiCloud we plan to distribute data onto geographically decentralized storage resources. Also we plan to add cloud-based web storage services. Necessary data replication is implemented for fault tolerance purpose.

**Infrastructure.** The underlying infrastructure of HiCloud is Eucalyptus, which provides interfaces in supporting customized cloud configuration on multiple sites. Current release of Eucalyptus runs on top of Xen [25] virtual machine, which utilizes private network established by Virtual Distributed Ethernet [26]. To get HiCloud running, every participating computing node should be equipped with Eucalyptus and configured accordingly.

### B. HiCloud Workflow

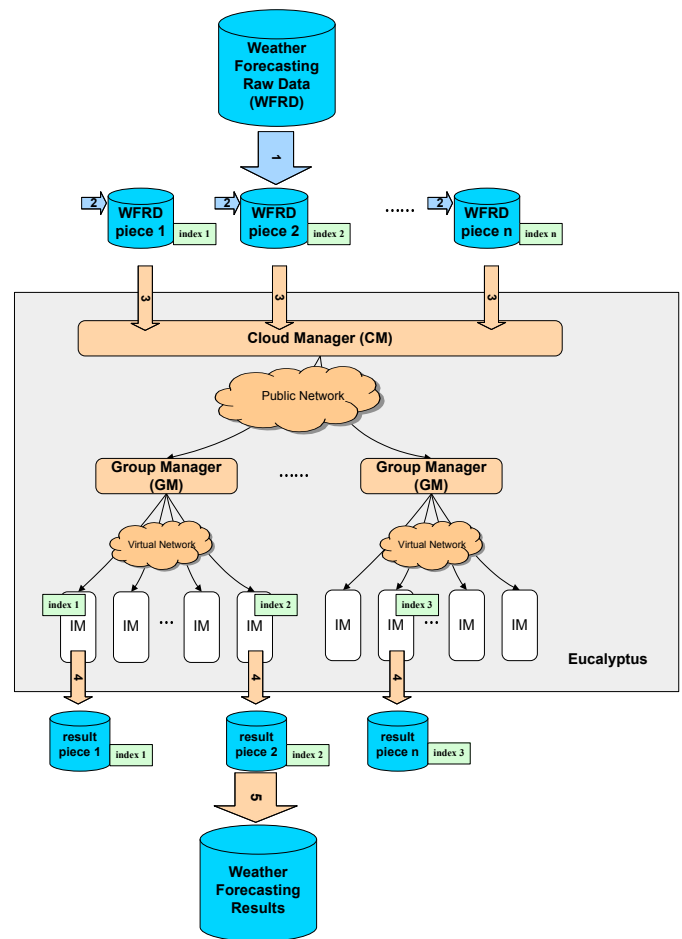


Fig. 4. HiCloud workflow. The labeled steps are: 1) data slicing; 2) data indexing; 3) task configuration; 4) task computation; 5) result assembly.

Figure 4 illustrates the data processing process in HiCloud. The workflow consists of five major steps as marked in the figure. Each of the five processes is elaborated below.

**Data slicing.** In data slicing raw data generated by weather forecasting applications is directed to specifically implemented programs and are sliced into chunks for further processing. Therefore it is crucial to define appropriate data partitioning

schemes for minimal communication overheads and workload balancing. One possible solution is to employ the map-reduce framework [27], [28] for independent raw data. Moreover our prior research has achieved positive results on this issue. For example, the adaptive hierarchical partitioning [29] and adaptive clustering [30] algorithms are effective in partitioning data into subdomains for specific requirements.

**Data Indexing.** Data indexing has two key functions. First it encapsulates property information for WFRD chunks and second, it generates unique data package to identify the chunk. There are two primary reasons of performing data indexing: 1) data index containing property information of WFRD chunk such as location, size and feature, is imperative for Instance Manager (IM) to locate WFRD chunk when it has been assigned computing tasks; 2) As data generated from HRWF is of tera-scale, it is generally impractical for Eucalyptus Cloud Manager (CM) to send a copy to the underlying IM since such behavior will result in overburden of public network. In HiCloud only data index is dispatched to and parsed by IMs. By doing so IM gets sufficient information for computing.

The format of index is defined as:

$$IDX = \{name, partition\_info, location, size, reserved, checksum\}$$

- *name*: Each index is assigned a name which is not necessarily unique.
- *partition\_info*: This field is used to locate position of one specific WFRD chunk in the whole data file.
- *location*: identifies location of computing launcher. For example, by using IP addresses, the field value of "139.78.1.1" means current WFRD chunk's computational task is requested by a workstation located at "139.78.1.1".
- *size*: indicates how large for one WFRD chunk.
- *reserved*: reserved for future use.
- *checksum*: Redundant field which guarantees index is right after traversal through public network.

**Task Configuration.** In HiCloud computational tasks are customized through request issuing. Request of HiCloud is encapsulated into WWW request or WS request which are supported by Eucalyptus directly. Once a request is generated it is dispatched to CM for reserving resources for future computing.

The format of request is defined as:

$$REQ = \{auth\_info, comp\_type, idx\}$$

*auth\_info* is used to identify computing launcher's authorization information. Any invalid authorization information will lead to an operation of issuing an authorization-fail message and rejection of service. The other two fields (Computing Type and Index) help CM to classify computing tasks.

**task computation.** As soon as the HiCloud request is satisfied and sufficient resources are allocated, the computing process starts. Independent GM running on front-end of each cluster manages local fabric layer consisting of physical

resources and starts dispatching computing tasks for each IM. The computing results are stored locally and will be used for later assembling. Upon finishing the computing IM issues a JOB-FIN message to notify its current status.

**result assembly.** After all IMs participating in computing process sending back JOB-FIN message CM will trigger assembling process. The process is customized, utilizing certain fields of index such as *partition\_info* and *location* fields to finalize current computing cycle.

#### IV. EXPECTED OUTCOME AND IMPACT

We will build the system and conduct extensive performance evaluation on a large-scale cloud consisting of three sites: OSU (Oklahoma State University), GWU (George Washington University), and OU (University of Oklahoma). OSU hosts a supercomputer with 512 quad-core CPUs, OU hosts a supercomputer with 1072 quad-core CPUs, and GWU hosts a supercomputer with 1048 quad-core CPUs. Future experiments will also be conducted on NSF TeraGrid and other national labs.

We expect the proposed research will result in the following impacts: (1) For the first time, the HWRF application is executed on multiple geographically-distributed sites at very large-scale. This breakthrough could introduce high-impact implication, e.g., a researcher who can only use a small set of computers now can use a very large pool of resources cross multiple sites, and idle resources can be pooled together to allow users' access with a familiar or conventional interface. (2) For the first time, cloud infrastructure (currently dominated by commercial vendors) is used for realworld large-scale scientific applications. and (3) Weather forecasting research can be forged forward to offer higher resolution of results conveniently.

#### V. SUMMARY

In summary, we propose a HiCloud framework to enable large-scale high resolution weather forecasting on clouds of multi-site of supercomputers with over 2500 CPUs. We expect to achieve breakthroughs on both weather forecasting applications with higher resolution and cloud infrastructure of multi-site supercomputers supporting realworld applications.

#### REFERENCES

- [1] B. Hayes, "Cloud computing," *Communication of the ACM*, vol. 51, no. 7, pp. 9–11, 2008.
- [2] R. Buyya, C. S. Yeo, and S. Venugopal, "Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities," *High Performance Computing and Communications, 2008. HPCC '08. 10th IEEE International Conference on*, pp. 5–13, Sept. 2008.
- [3] I. Foster, Y. Zhao, I. Raicu, and S. Lu, "Cloud computing and grid computing 360-degree compared," in *Grid Computing Environments Workshop, 2008. GCE '08, 2008*, pp. 1–10.
- [4] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "Above the clouds: A Berkeley view of cloud computing," EERC Department, University of California, Berkeley, Tech. Rep. UCB/EERC-2009-28, Feb 2009.
- [5] "The Internet Cloud," Website, <http://www.thestandard.com/article/0,1902,5466,00.html>.

- [6] M. Xue, F. Kong, K. W. Thomas, J. Gao, Y. Wang, K. Brewster, K. K. Droegemeier, J. S. Kain, S. J. Weiss, D. R. Bright, M. C. Coniglio, and J. Du, "Extended abstracts: Caps realtime storm-scale ensemble and high-resolution forecasts as part of the noaa hazardous weather testbed 2008 spring experiment," in *24th Conf. on Severe Local Storms*, oct.
- [7] F. Kong, M. Xue, K. W. Thomas, K. K. Droegemeier, Y. Wang, K. Brewster, J. Gao, J. S. Kain, S. J. Weiss, D. R. Bright, M. C. Coniglio, and J. Du, "Preprints: Real-time storm-scale ensemble forecast experiment-analysis of 2008 spring experiment data," in *24th Conf. on Severe Local Storms*, oct.
- [8] "Wikipedia: Cloud Computing," Website, [http://en.wikipedia.org/wiki/Cloud\\_computing](http://en.wikipedia.org/wiki/Cloud_computing).
- [9] "Cloud Taxonomy," Website, <http://peterlaird.blogspot.com/2008/09/visual-map-of-cloud-computingsaaspaas.html>.
- [10] I. Foster and C. Kesselman, Eds., *The Grid: Blueprint for a New Computing Infrastructure*. San Francisco, CA: Morgan Kaufmann, 1999.
- [11] I. Foster, C. Kesselman, J. Nick, and S. Tuecke, "The physiology of the grid: An open grid services architecture for distributed systems integration, open grid service infrastructure wg, global grid forum," June 2002.
- [12] I. Foster and C. Kesselman, *The Grid: Blueprint for a New Computing Infrastructure*, 2nd ed. Morgan Kaufmann, 2004.
- [13] "Whatis.com: What is Cloud Computing?" Website, [http://searchenterprisedesktop.techtarget.com/sDefinition/0,,sid192\\_gci1287881,00.html](http://searchenterprisedesktop.techtarget.com/sDefinition/0,,sid192_gci1287881,00.html).
- [14] "GoogleAppEngine," Website, <http://code.google.com/appengine/>.
- [15] "Amazon EC2," Website, <http://aws.amazon.com/ec2/>.
- [16] D. Nurmi, R. Wolski, C. Grzegorzcyk, G. Obertelli, S. Soman, L. Youseff, and D. Zagorodnov, "The eucalyptus open-source cloud-computing system," in *in Proceedings of Cloud Computing and Its Applications*, 2008.
- [17] C. Grzegorzcyk, D. Nurmi, G. Obertelliand, S. Rajagopalan, S. Soman, L. Youseff, and D. Zagorodnov, "Eucalyptus: An elastic utility computing architecture for linking your programs to useful systems (presentation)," in *Open Source Grid and Cluster'08*, 2008.
- [18] C. Hoffa, G. Mehta, T. Freeman, E. Deelman, K. Keahey, B. Berriman, and J. Good, "On the use of cloud computing for scientific workflows," *IEEE International Conference on eScience*, pp. 640–645, 2008.
- [19] "Enomalism," Website, <http://www.enomaly.com/>.
- [20] S. Ghemawat, H. Gobioff, and S.-T. Leung, "The google file system," in *SOSP '03: Proceedings of the nineteenth ACM symposium on Operating systems principles*. ACM Press, 2003, pp. 29–43.
- [21] H. Liu and D. Orban, "Gridbatch: Cloud computing for large-scale data-intensive batch applications," 2008, pp. 295–305.
- [22] "Cloud Computing Stack," Website, <http://samj.net/2008/09/taxonomy-6-layer-cloud-computing-stack.html>.
- [23] "Apache Axis2," Website, <http://ws.apache.org/axis2/>.
- [24] "Jetty," Website, <http://www.mortbay.org/jetty/>.
- [25] "Xen," Website, <http://www.xen.org/>.
- [26] "Virtual Distributed Ethernet," Website, <http://vde.sourceforge.net/>.
- [27] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Communication of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.
- [28] "Hadoop," Website, <http://hadoop.apache.org/core/>.
- [29] X. Li and M. Parashar, "Hierarchical partitioning techniques for structured adaptive mesh refinement applications," *The Journal of Supercomputing*, vol. 28, no. 3, pp. 265–278, 2004.
- [30] ———, "Adaptive runtime management of spatial and temporal heterogeneity for dynamic grid applications," in *Proceedings of the 13th High Performance Computing Symposium (HPC)*, 2005, pp. 223–228.