Future Generation Computer Systems 27 (2011) 440–453

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Hybrid Computing—Where HPC meets grid and Cloud Computing

Gabriel Mateescu [a,*], Wolfgang Gentzsch [b], Calvin J. Ribbens [c]

[a] National Center for Supercomputing Applications, Urbana-Champaign, IL 61801, USA
[b] Open Grid Forum, P.O. Box 2326, Joliet, IL 60434, USA
[c] Computer Science Department, Virginia Tech, Blacksburg, VA 24061, USA

ARTICLE INFO

Article history:
Received 2 July 2010
Received in revised form
25 October 2010
Accepted 3 November 2010
Available online 26 November 2010

Keywords:
Distributed systems
Scheduling
Cloud computing
HPC
Statistical reservation
Cloud bursting
Virtualization
Distributed hash table

ABSTRACT

We introduce a hybrid High Performance Computing (HPC) infrastructure architecture that provides predictable execution of scientific applications, and scales from a single resource to multiple resources, with different ownership, policy, and geographic locations. We identify three paradigms in the evolution of HPC and high-throughput computing: owner-centric HPC (traditional), Grid computing, and Cloud computing. After analyzing the synergies among HPC, Grid and Cloud computing, we argue for an architecture that combines the benefits of these technologies. We call the building block of this architecture, Elastic Cluster. We describe the concept of Elastic Cluster and show how it can be used to achieve effective and predictable execution of HPC workloads. Then we discuss implementation aspects, and propose a new distributed information system design that combines features of distributed hash tables and relational databases.

1. Introduction

High Performance Computing (HPC) has a long tradition of leveraging trends and technologies from the broader computing community to yield greater computational power for solving important computational science and engineering problems. While many computer architecture innovations were first motivated by HPC, many other advances in HPC flowed in the other direction, i.e., from the broader computing community to the world of HPC and scientific applications. For example, like everyone else, HPC system designers and users have benefited from the decades-long advance of Moore's law. Similarly, commodity microprocessors developed and marketed for the desktop have played a key role in the rise of large-scale parallel clusters. Recently, graphical processing units (GPUs) have become a major component of modern HPC systems. Even Grid computing, which in its infancy was motivated in large parts by scientific workflows, has become a much broader technology with widespread application in a variety of non-HPC contexts.

In this paper, we examine how another broad development in computing, namely Cloud computing, can best be combined with traditional HPC approaches to offer greater problem-solving resources for scientific workflows [1]. In fact, since there are important features of Grid computing that complement HPC and Cloud computing, we will think through the relationships among these three paradigms – traditional HPC, Grid and Cloud – in terms of their motivation, strengths and weaknesses. We seek to combine the best attributes of each, and propose a model for how these paradigms can work together and how scientific workloads can be managed in such a *hybrid computing* environment.

Our key contributions are the following: (1) an architecture for hybrid computing that supports functionality not found in other application execution systems; (2) a mechanism for providing statistical resource reservations for batch jobs; and (3) a distributed information system design that achieves guaranteed data discovery, scalability, and fault resilience. The cornerstone of the proposed hybrid computing architecture is the *Elastic Cluster*, a model of self-resizable computing resources that make up the emerging hybrid computing environment.

The remainder of the paper is organized as follows. Section 2 identifies the defining attributes of traditional HPC, Grid and Cloud computing, and examines the benefits of combining the strengths of each paradigm. In Section 3, we introduce the concept of Elastic Cluster as a general model of the computing resources that make up the emerging hybrid computing environment. Then, in Section 4, we show how an Elastic Cluster can be used to build a large-scale workflow management system. After discussing implementation aspects of the Elastic Cluster in Section 5, we introduce in Section 6 a new design, called key-partitioned database, for the

* Corresponding author. Tel.: +1 540 339 3633; fax: +1 217 244 9757.
E-mail address: mateescu@acm.org (G. Mateescu).

0167-739X/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.future.2010.11.003

distributed information systems that are components of the proposed architecture. We survey related work in Section 7, and give concluding remarks in Section 8.

2. Convergence of HPC, Grid and Cloud computing

2.1. Attributes

Today's hybrid computing ecosystem represents the intersection of three broad paradigms for computing infrastructure and use: (1) Owner-centric (traditional) HPC; (2) Grid computing (resource sharing); (3) Cloud computing (on-demand resource/service provisioning).

Each paradigm is characterized by a set of attributes of the *resources* making up the infrastructure and of the *applications* executing on that infrastructure. These attributes include: (1) *resource ownership*: either locally owned resources or externally owned resources; (2) *resource accessibility*: either private (only by the owner of the resource) or public (available to entities other than the owner); (3) *resource sizing*: either *quasi-static* (resources grow when purchasing new hardware) or *dynamic* (resources can grow dynamically by using external, public, resources); (4) *resource allocation policy*: either exclusive (per-organization, or per-group, or per-project, or per-user) or shared among organizations, groups, projects, or users; (5) *application portability*: either tied to a specific platform (e.g., hardware, operating system) or platform-agnostic (easily portable). In terms of these attributes, we describe next the three paradigms that combine to support hybrid computing.

In *Owner-centric (traditional) HPC*, resources are locally owned, with private access (for members of the owner organization and its partners); resources have quasi-static size that changes by purchasing new resources. Allocation is typically exclusive with respect to a group, either shared or exclusive with respect to users and projects, and shared with respect to the workflows of a user or project. HPC applications have evolved from platform-specific to portable.

In *Grid computing*, resources are both locally and externally owned; accessing external resources provides additional capacity and capability, and the mixed blessing of accessing heterogeneous resources. Resource access is public, meaning that some of the local resources are made available to external users who are members of a certain *Virtual Organization* [2]. Resource size is dynamic, growing by way of accessing external, public resources. Allocation builds on the methods used in traditional HPC to differentiate between local and external users; in the case of free-of-charge access, external (Grid) users often get lower quality of service than local users. With fee-based access, the Grid provider supports the type of allocation specified in a service agreement. Grid applications are a mix of portable and platform-specific, with platform dependencies for the codes that exploit leading-edge HPC architectures. Well-known Grid computing projects include the Globus Toolkit [3,4], UNICORE [5] and the NSF TeraGrid [6] infrastructure.

In *Cloud computing* [7], resources can be either externally owned (public cloud), or internally owned (private cloud), the former being offered by Cloud providers. Public clouds offer access to external users who are typically billed on a pay-as-you-use basis. Resource size is dynamic, growing by way of on-demand creation of the resources of the desired type (e.g., virtual machines), this kind of dynamic sizing being supported by virtualization technologies that enable dynamic creation, migration, and destruction of resources; The *Infrastructure as a Service* (IaaS) offered by Cloud providers is well suited for HPC workloads. IaaS services include virtual machines and the management interface (for example, the Amazon Elastic Compute Cloud (EC2) [8]), as well as virtual storage (for example, the Amazon Simple Storage Service (S3) [9]).

Attribute	HPC	Grid	Cloud
Capacity	fixed	average to high; growth by aggregating independently managed resources	**high; growth by elasticity of commonly managed resources**
Capability	**very high**	average to high	low to average
Virtual Machine Support	rarely	sometimes	**always**
Resource sharing	limited	**high**	limited
Resource heterogeneity	low	**average to high**	low to average
Built-in Workload Management	**yes**	**yes**	no
Distribute Workload Across Resources from Multiple Admin Domains	not applicable	**yes**	no
Interoperability	not applicable	**average**	low
Security	**high**	average	low to average

Fig. 1. Comparison of key attributes of traditional HPC, Grid, and Cloud.

2.2. Combining the paradigms

Each of the three major computing paradigms has its strengths and weaknesses. The motivation for hybrid computing is to combine all three paradigms so that strengths are maintained or even enhanced, and weakness are reduced. The strengths and weakness of each paradigm are well known and much studied. It is not our purpose here to provide a comprehensive evaluation of each individual approach. However, even a high-level summary of the attributes of each paradigm helps identify areas where combining approaches shows promise. Fig. 1 summarizes some key attributes of the HPC, Grid and Cloud approaches. We note that no single paradigm is the best solution from all points of view.

For example, there are important differences in the three paradigms with respect to the "capacity vs. capability" distinction. *Capability* resources are needed for demanding applications, such as tightly coupled, highly parallel codes or large shared memory applications. *Capacity* resources are well suited for throughput-intensive applications, which consist of many loosely coupled (or independent) processes. A capacity resource is typically an installation with commodity components (servers, storage, interconnect); a capability resource is an HPC installation with special hardware, such as low-latency interconnects (e.g., InfiniBand), storage-area networks, many-core nodes, or nodes with hundreds of Gigabytes of main memory.

The traditional, owner-centric HPC paradigm excels in handling capability workloads in a well-managed, secure environment. However, capacity is fixed in this domain, and there is typically weak support for virtualization and resource sharing.

Strengths of Grid computing include access to additional HPC capacity and capability, which also promotes better utilization of resources. Grid computing enables exposing heterogeneous resources with a unified interface, thereby allowing users to access multiple resources in a uniform manner. However, Grids have limited interoperability between different Grid software stacks. Moreover, the use of an external resource on a non-fee basis is often associated with limited expectations about the reliability and long-term availability of a resource. While Grids enable a wider choice of resources, the amount of resources of each type is still constant within a non-virtualized Grid.

A key strength of Cloud computing is on-demand resource provisioning (thanks to virtualization) which enables capacity resizing, workload migration, better availability and performance. Clouds lag in interoperability (which limits resource sharing across Cloud providers), security, and built-in workload management services.