Future Generation Computer Systems 37 (2014) 148–161

Contents lists available at ScienceDirect

Future Generation Computer Systems

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

Adaptive parallel application resource remapping through the live migration of virtual machines

Muhammad Atif[a,*], Peter Strazdins[b]

[a] National Computational Infrastructure, The Australian National University, Canberra, ACT, 0200, Australia

[b] Department of Computer Science, College of Engineering and Computer Science, The Australian National University, Canberra, ACT, 0200, Australia

HIGHLIGHTS

- We present a framework which deals with the issue of heterogeneity in clusters.
- Mathematical Model for performance prediction and migration of job within clusters.
- The framework is analyzed with comprehensive set of experiments (no simulations).
- Sensitivity analysis is provided to determine the impact of various parameters.
- Improve the throughput of a moderately heterogeneous compute farm by up to 25%.

ARTICLE INFO

Article history:
Received 15 April 2012
Received in revised form
14 May 2013
Accepted 28 June 2013
Available online 19 July 2013

Keywords:
Performance prediction
Virtualization
Heterogeneous clusters
Live migration
Cluster scheduling
Resource management

ABSTRACT

In this paper we present ARRIVE-F, a novel open source framework which addresses the issue of heterogeneity in virtualized compute farms, such as those hosted by a cloud infrastructure provider. Unlike the previous attempts, our framework is not based on linear frequency models and does not require source code modifications or off-line profiling. The heterogeneous compute farm is first divided into a number of homogeneous sub-clusters. The framework then carries out a lightweight 'online' profiling of the CPU, communication and memory subsystems of all the active jobs in the compute farm. From this, it constructs a performance model to predict the execution times of each job on all the distinct sub-clusters in the compute farm. Based upon the predicted execution times, the framework is able to relocate the compute jobs to the currently best-suited hardware platforms such that the overall throughput of the compute farm is increased. We utilize the live migration feature of virtual machine monitors to migrate the job from one sub-cluster to another.

The prediction accuracy of our performance estimation model is over 80%. The implementation of ARRIVE-F is lightweight, with an overhead of 3%. Experiments on a synthetic workload of scientific benchmarks show that we are able to improve the throughput of a moderately heterogeneous compute farm by up to 25%, with a time saving of up to 33%.

1. Introduction

Compute farms, whether for research department clusters, data centers, cloud infrastructure providers or supercomputing facilities tend to become heterogeneous over time. This is due to incremental extension over a period of time and/or particular nodes being purchased for users with particular needs and the relatively small price differences between these various options. After couple of upgrade cycles, the compute farm becomes a heterogeneous compute farm (HC) constituted of a federation of homogeneous sub-clusters.

Parallel applications have varied computation and communication requirements depending on the domain and the nature of their algorithms. For instance, some applications are floating point intensive while others can be memory or communication intensive. This diverse nature of applications results in varied execution time in the compute farm due to the heterogeneity of the nodes.

On a heterogeneous cluster where only half of the nodes are linked via an expensive high performance network, it is possible for a parallel application that does little inter-node communication to end up running on the nodes with the fast network, while another application that would greatly benefit from that network is left to run on a slower half of the cluster.

The issue of effective mapping (scheduling) of parallel applications onto such heterogeneous systems is therefore of great interest to researchers. The problem is NP-complete [1] and several research studies have addressed this problem by developing

* Corresponding author. Tel.: +61 430 393863.
E-mail addresses: muhammad.atif@anu.edu.au (M. Atif), peter.strazdins@anu.edu.au (P. Strazdins).

0167-739X/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.future.2013.06.028

heuristic techniques [1,2]. These heuristics require the scheduler to be aware of the application characteristics and are static in nature. This in turn requires the application programmer to profile and analyze the application prior to job submission. The static nature of this approach prevents dynamic load balancing within the compute farm.

Another approach to address the computation and communication imbalance of the nodes in an HC is to adapt the parallel application according to the heterogeneity of the compute farm. Here, the parallel application is required to distribute computations unevenly to account for the varied speed and architecture of processors [3,4]. The load balancing of such systems require a considerable effort from the programmers perspective [5] and the solutions are not generic in nature. In the case of workload, source code changes are required, which is difficult and time consuming. In order to load balance an application, the programmer is required to determine the application's performance characteristics. In both cases, the allocation of nodes to a parallel job in an HC requires some sort of performance modeling techniques. In performance modeling, an application is profiled to gain an understanding of its performance characteristics. The performance models are evaluated on the different compute nodes and sub-networks to determine the expected speedups (or slowdowns) on various hardware architectures/nodes.

Fine grained performance modeling is capable of reasonably accurate prediction but the associated cost of profiling can be very high in terms of the wall-clock time of the job [4,5]. Due to these costs, these techniques must be applied to applications in an 'offline' mode. The application or some of its iterations are profiled on a set of hardware and scheduling decisions are based on the resultant profile metrics. In the case of a workload change, the application behavior changes and the application needs to be profiled again.

In this paper, we present the design and results of our resource remapping framework, which is able to exploit the heterogeneity in a compute farm to improve throughput. We deal with this issue of heterogeneity by breaking the heterogeneous compute farm into a number of homogeneous sub-clusters. The runtime characteristics of applications are determined with the combination of hardware performance counters/units (PMUs) and the profiling the underlying communication and I/O library, in this case the profiling interface to MPI (PMPI). This enables us to predict the performance of the running MPI applications on all the other sub-clusters present in our heterogeneous compute farm. All this is done without the need of changing the application binary or requiring off-line profiling and analysis. We then propose the best-suited sub-cluster for the compute job on the compute farm and migrate the job to this sub-cluster to improve the overall throughput and the average waiting time. For the job migration (or remapping), we make use of the live migration facility provided by most virtual machine monitors. As concluded in [6,7], the benefits of virtualization in HPC environments outweigh the potential overheads, namely potentially slower communication and CPU overhead due to virtualization. In the absence of virtualization, one can use the checkpoint and restart facilities [8] in MPI implementations.

The rest of the paper is organized as follows: We briefly discuss the related work in Section 2. We then present our theoretical framework in Section 3. Implementation details are provided in Section 4. Sections 5 and 6 detail our experiments and results. Conclusions and future work are given in Section 9.

2. Related work

There is a significant amount of work related to this paper which can be broadly divided into two categories: heterogeneity-aware schedulers and heterogeneity-aware applications.

2.1. Heterogeneity-aware schedulers

A number of 'static' heterogeneous cluster scheduling solutions have been proposed which try to map the application processes on the available compute nodes to improve the application's runtime. The general problem of optimally mapping parallel programs or tasks to machines in a heterogeneous compute farm has been shown to be NP-complete [1], and hence requires heuristics [1,9] Braun et al. [2] have presented a detailed comparison of a number of heuristics proposed earlier.[1] However these heuristics have a hard requirement of the estimated execution times of the application on all the possible distinct compute nodes (processors). This either requires the complete execution of the application on all the possible compute nodes before scheduling or 'offline' performance estimation to predict the runtime of the application. All of the subsequent work in the area is static in nature and dynamic load balancing of the HC is missing in these frameworks.

Hsu et al. [10] have proposed Cloud Adaptive Dispatching (CAD) for improving the performance of data intensive applications in the heterogeneous cloud environment. CAD is able to arrange inter-cluster communications across heterogeneous networks by utilizing the Local Messages Reduction (LMR) and Remote Message Amplification (RMA) techniques. It is shown through simulations that CAD can significantly improve the communication costs by scheduling messages according to the characteristic of messages.

Katramatos et al. have proposed Cost/Benefit Estimating Service (CBES) [4], the core of which is the hardware mapping operation which compares and select the most beneficial hardware mapping at the given time. CBES consists of a set of databases, monitoring services and profiling tools divided into two categories: system-dedicated and application-dedicated. The system-dedicated infrastructure is an 'off-line' phase and is used to create a profile of the computing system. Once a system profile is generated, the system continuously monitors the resource status (load information). The application-dedicated infrastructure is responsible for generating communication and computation profiles of the application. However, CBES bases its performance estimates on CPU frequency and it cannot move jobs between the clusters once dispatched.

2.2. Heterogeneity-aware applications

One approach proposed by several researchers [3,11] to address the computation and communication imbalance of the nodes in an HC is to adapt the parallel application according to the heterogeneity of the compute farm. Here, the parallel application is required to distribute computations unevenly to account for the varied speed and architecture of processors [3]. These techniques require source-code modifications. Charm++ [11], Prophet [12], AppLeS [13] and EasyGrid [14] are examples of similar attempts that require source code modifications to build scheduling and load balancing capabilities inside an application.

Charm++ [11] is a machine independent parallel programming system. Charm++ programs are written in C++ with a few library calls and an interface description language for publishing Charm++ objects. Charm++supports dynamic load balancing using object migration for irregular and dynamic applications. To support MPI, Charm++ has its own layer known as AMPI (Adaptive MPI). However, it requires developers to rewrite MPI applications using Charm++ language specifications. AMPI is not MPI-2 complaint and requires the application developer to add support for dynamic load balancing in the application. This requires the

[1] Details and original citations of these heuristics can be found in the paper.