클라우드컴퓨팅에서 메시지패싱방식 응용프로그램의 효율적인 체크포인트 알고리즘

Duc Tai Le, Manh Thuong Quan Dao, Minjoon Ahn, and Hyunseung Choo
성균관대학교 정보통신공학부
e-mail : {ldtai, dmtquan, ahn.m.j, choo}@skku.edu

Efficient Checkpoint Algorithm for Message-Passing Parallel Applications on Cloud Computing

Duc Tai Le, Manh Thuong Quan Dao, Minjoon Ahn, and Huynseung Choo
School of Information and Communication Engineering, Sungkyunkwan University

Abstract

In this work, we study the checkpoint/restart problem for message-passing parallel applications running on cloud computing environment. This is a new direction which arises from the trend of enabling the applications to run on the cloud computing environment. The main objective is to propose an efficient checkpoint algorithm for message-passing parallel applications considering communications with external systems. We further implement the novel algorithm by modifying gSOAP and OpenMPI (the open source libraries) which support service calls and checkpoint message-passing parallel programs, especially. The simulation showed that additional costs to the executing and checkpointing application of the algorithm are negligible. Ultimately, the algorithm supports efficiently the checkpoint/restart service for message-passing parallel applications, that send requests to external services.

1. Introduction

Computing is being transformed to a model consisting of services that are commoditized and delivered in a manner similar to traditional utilities such as water, electricity, gas, and telephony. In such a model, applications reside in massively-scalable data centers, where computing resources can be dynamically provisioned and shared to achieve significant economies of scale, users access services based on their requirements through their connected devices. Several computing paradigms have promised to deliver this utility computing vision. These include Cluster Computing, Grid computing, and more recently Cloud computing [1]. The proliferation of smart mobile devices, high speed wireless connectivity, and rich browser-based Web 2.0 interfaces have made the network-based cloud computing model not only practical but also a source of reduced IT complexity. The strength of a cloud is its infrastructure management, enabled by the maturity and progress of virtualization technology to manage and better utilize the underlying resources through automatic provisioning, reimaging, workload rebalancing, monitoring, and systematic change request handling. These advantages of cloud provide a favorable environment for running many kinds of applications.

Consequently, the number of applications, that is ported to a cloud platform, developed on a cloud platform, or hosted on a cloud infrastructure, increases quickly. Therefore, the techniques to ensure safety or to reduce the cost of recovery when applications running on the cloud computing environment encounter some unexpected problems, such as hardware failure, power cut off, transmission lines disconnected, etc., are very necessary.

In this paper, the main objective is to propose an efficient checkpoint algorithm for message-passing parallel applications in the cloud computing environment.

2. Checkpoint/restart problem on cloud

Checkpoint/restart systems are used to provide system layer support for checkpoint and restart fault tolerance techniques. These systems capture an image (or a snapshot) of running process state and preserve it for later recovery. In the case of parallel application, checkpoint/restart coordination protocols generate a global snapshot by taking the union of all individual process snapshots, as illustrated by in Fig.1 [2].

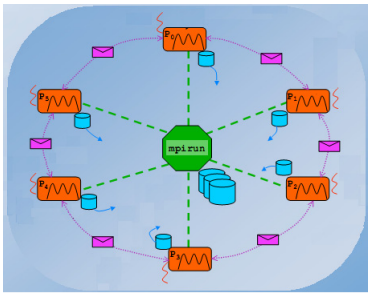

Fig.1. Checkpoint process of parallel application

On the cloud computing environment, there are a lot of available services around the application's execution environment. The user application tends to use these services rather than implement the equivalent functions. To simplify

the presentation of how the system communicates with the outside world, we model this world as a special process, called the "outside world process" (OWP), which interacts with the rest of the system. This special process cannot fail. Its state cannot be maintained and it cannot participate in the recovery protocol. However, when the error occurred and the system is restored to a certain state, some unexpected situations related to the OWP may happen. Therefore, the OWP process is always a big challenge for checkpoint/restart algorithms. Existing solutions which support checkpoint/restart message-passing parallel applications are ineffective in cloud. Therefore, a new checkpoint algorithm is strongly needed to solve this problem.

3. System model and assumptions

Applications can be transferred to running on cloud through the web portal, command line interface or APIs. Then, depending on the user requirements (agreed with the providers), these applications will be coordinated to the most suitable system in the cloud, as described in Fig.2. With advantage conditions are supported in the new environment, programmers can use many services, which are available on the cloud, by implementing a simple service call instead of having to rewrite similar functions directly in their applications.

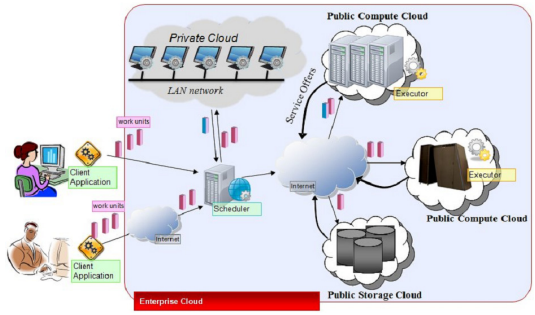

Fig.2. Applications running on enterprise cloud

In this paper, our OWP are considered as the stateless services. The same requirements, arising from the clients at any time, always get the same results. We also assume that the client uses only synchronous call to send the request for OWP. After sending the request, the running process will be temporarily locked to wait until receiving responses.

4. Proposed algorithm

We only consider the processes that are waiting the responses from OWP at the checkpoint time. To restore successfully the parallel application back to the previous checkpoint, we turn back these processes to the state before sending the request.

For example, considering a parallel application that has 2 processes $\{P_0, P_1\}$, as shown in Fig.3. Before sending a request to OWP, process P_0 will call WAIT_OWP function to mark that it is waiting responses. At the checkpoint time, the POST_CHECKPOINT function will mark all processes waiting the responses from OWP as uncompleted processes. When restoring application, the RELEASE_OWP function will turn these uncompleted processes back to the time before sending the request.

5. Simulation result

We implement the proposed algorithm by modifying two open source libraries: OpenMPI and gSOAP. These are the best rated libraries in supporting implementation calls to a service (gSOAP) as well as checkpoint for the message-passing parallel programs (OpenMPI). We measure the delay time of prime counting parallel application to evaluate the additional cost of our proposed scheme.

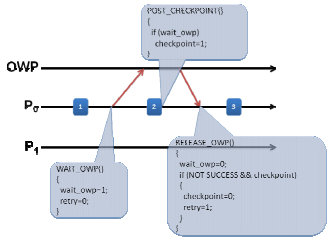

Fig.3. Scheme overview

As shown in Fig.4, the delay time of our proposed scheme is close to the original scheme. However, our proposed scheme can guarantee that the probability of successful state restoring is 100%.

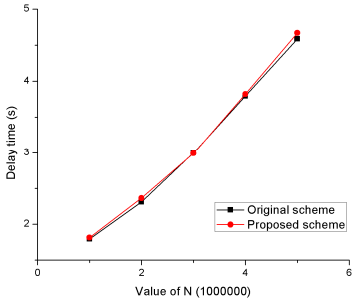

Fig.4. Delay time of prime counting application

6. Conclusion

In this paper, we proposed an efficient checkpoint algorithm for message-passing parallel applications on cloud computing. The proposed algorithm can be easily integrated into OpenMPI and gSOAP and other open source libraries. The simulation results show that, our proposed scheme can guarantee 100% successful state restoring with negligible additional costs.

ACKNOWLEDGMENT

This research was supported in part by MKE and MEST, Korean government, under ITRC NIPA-2011-(C1090-1121-0008), WCU NRF (No. R31-2010-000-10062-0) and PRCP(2010-0020210) through NRF, respectively.

References

[1] Vaquero LM., Rodero-Merino L., Cáceres J., Lindner M. "A Break in the Clouds: Towards a Cloud Definition", ACM Computer Communication Reviews, January 2009

[2] Joshua Hursey, Jeffrey M. Squyres, Andrew Lumsdaine, A Checkpoint and Restart Service Specification for Open MPI, Indiana University, Bloomington, Indiana, USA, July 2007, http://www.cs.indiana.edu/cgi-bin/techreports/TRNNN.cgi?trnum=TR635