論文 / 著書情報
Article / Book Information

題目(和文)	最適なチェックポ　イント・リスタートのための設計及び　実装
Title(English)	Design and Implementation for Optimal Checkpoint/Restart
著者(和文)	佐藤賢斗
Author(English)	Kento Sato
出典(和文)	学位:博士(理学), 学位授与機関:東京工業大学, 報告番号:甲第9422号, 授与年月日:2014年3月26日, 学位の種別:課程博士, 審査員:松岡　聡,遠藤　敏夫,増原　英彦,三好　直人,脇田　建
Citation(English)	Degree:Doctor (Science), Conferring organization: Tokyo Institute of Technology, Report number:甲第9422号, Conferred date:2014/3/26, Degree Type:Course doctor, Examiner:,,,,
学位種別(和文)	博士論文
Category(English)	Doctoral Thesis
種別(和文)	論文要旨
Type(English)	Summary

（博士課程）
Doctoral Program

論 文 要 旨

THESIS SUMMARY

専攻： Department of	数理・計算科学　専攻	申請学位（専攻分野）： Academic Degree Requested	博士 Doctor of Science　（理学）
学生氏名： Student's Name	佐藤　賢斗	指導教員（主）： Academic Advisor(main)	教授　松岡　聡
		指導教員（副）： Academic Advisor(sub)	

要旨（英文 800 語程度）
Thesis Summary（approx.800 English Words ）

The growing computational power of high performance computing (HPC) systems enables increasingly larger scientific simulations. However, as the number of system components increase, the overall failure rate of systems increases. Further, the mean time between failures (MTBF) of future systems is projected to be on the order of tens of minutes or hours. In fact, an earlier failure analysis on Hera, Atlas and Coastal clusters at Lawrence Livermore National Laboratory (LLNL) showed that a production application, the pF3D laser-plasma interaction code, experienced 191 failures out of 5-million node-hours. If we simply scale out the system while keeping the failure rate constant, the estimated MTBF is about 1.2 days for a 1,000-node cluster, 2.9 hours for a 10,000-node cluster, and 17 minutes for a 100,000-node cluster. Without fault tolerant techniques and more reliable hardware, applications will be unable to run continuously for even one day on such a larger system. Therefore, as we look towards extreme scale systems, fault tolerance is becoming more important.

Checkpointing is one of indispensable fault tolerance techniques, commonly used by HPC applications that run continuously for hours or days at a time. A checkpoint is a snapshot of application state that can be used to restart execution if a failure occurs. However, when checkpointing large-scale systems, tens of thousands of compute nodes write checkpoints to a parallel file system (PFS) concurrently, and the low I/O throughput becomes a bottleneck. Although simple, this straightforward checkpointing scheme can impose huge overheads on application run times.

To solve the problems, we develop asynchronous checkpointing system for light-weight checkpointing to PFS, explore multi-tier storage design with burst buffer for reliable checkpointing, and propose fault tolerant messaging interface (FMI) for fast and transparent recovery.

First, we develop an asynchronous checkpointing system to minimize checkpointing overhead/workload to PFS. The asynchronous checkpointing system solves the problem through agents running on additional nodes that asynchronously transfer checkpoints from the compute nodes to the PFS. Our approach has two key advantages. It lowers application checkpoint overhead by overlapping computation and writing checkpoints to the PFS. Also, it reduces PFS load by using fewer concurrent writers and moderating the rate of PFS I/O operations. In particular, our asynchronous checkpointing system coupled with a multi-level checkpoint/restart technique maintains a given application efficiency with significantly lower PFS requirements than simple synchronous checkpointing, which write checkpoints such that all processes write their own checkpoints concurrently, and are blocked until the checkpoint operation completes.

Second, we explore multi-tier storage design for resilient architecture using burst buffers. Burst buffers have been proposed to alleviate the problems of I/O operations to a shared PFS. A Burst buffer is a new tier in the storage hierarchy to fill the performance gap between node-local storage and the PFS, and is shared by a subset of compute nodes. The new tire can absorb the bursty I/O requests from applications and thus can reduce the effective load on the PFS. We consider using burst buffers from different viewpoint, and try to improve system resiliency with burst buffer storage design. With burst buffers, an application can store checkpoints on a smaller number of dedicated burst buffer nodes, so the probability of lost checkpoints is decreased. We explore how burst buffers can improve efficiency and resiliency compared to using node-local storage instead of burst buffers based on a performance model combined with our multi-level asynchronous checkpoint/restart model. From our exploration, we found that burst buffers are indeed beneficial for checkpoint/restart on future systems, increasing reliability and efficiency.

Third, we propose FMI for fast and transparent recovery. FMI is a survivable messaging interface that uses fast, transparent in-memory checkpoint/restart and dynamic node allocation. With FMI, a developer writes an application using semantics similar to Message Passing Interface (MPI). The FMI runtime ensures that the application runs through failures by handling the activities needed for fault tolerance, such as checkpointing, failure detections, and recoveries. All of this motivates the need for a survivable messaging runtime system. Such a system should be able to maintain processes and connections that are unaffected by the failure while starting and integrating replacement processes as needed. Our implementation of FMI has performance comparable to MPI. Our experiments with a Poisson equation solver show that running with FMI incurs only a 28% overhead with a very high MTBF of 1 minute. By defining a simplified programming model and custom runtime, we find that FMI significantly improves resilience overheads compared to similar existing multi-level checkpointing methods and MPI implementations.

As a whole, these approaches enable fast, scalable and transparent checkpoint and restart. Our systems are expected to be indispensible for extreme scale systems.

備考 ： 論文要旨は、和文 2000 字と英文 300 語を 1 部ずつ提出するか、もしくは英文 800 語を 1 部提出してください。

Note : Thesis Summary should be submitted in either a copy of 2000 Japanese Characters and 300 Words (English) or 1copy of 800 Words (English).