분산 RAID 기반의 클러스터 시스템을 위한 분할된 결함허용정보 저장 기법

장 윤 석†

요 약

본 논문에서는 서버를 사용하지 않고 각 노드에 연결된 지역 디스크들을 이용하여 분산 RAID 저장 장치를 구성하는 분산 환경의 클러스터 컴퓨터를 위한 분할된 결함허용정보 저장 기법을 제안한다. 클러스터 노드들의 결함허용정보를 주기적으로 동시에 분산 RAID에 저장하는 방법은 분산 RAID의 병렬성을 이용할 수 있고, 각 노드의 결함허용정보를 순차적으로 저장하는 기법은 분산 환경에서 네트워크에 병목 현상을 방지할 수 있는 장점을 가지고 있다. 본 연구에서는 분산 RAID를 저장 장치로 사용하는 클러스터 컴퓨터에서 이들 두 가지 기법을 결합함으로써 통신 부하가 큰 응용에서 노드들에 대한 결함허용정보 저장 비용을 줄이고 클러스터의 가용성을 높일 수 있도록 하였다. 제안된 기법의 성능을 검증하기 위하여 본 연구에서는 16노드의 클러스터 시스템에서 MPI와 Linpack HPC 벤치마크 프로그램을 이용한 성능 평가를 수행하였다. 벤치마크 결과는 분할된 결함허용정보 저장 기법이 기존의 기법들에 비하여 분산 RAID를 사용한 클러스터 컴퓨터에서 비교적 우수한 성능을 나타낼 수 있으며, 클러스터의 단일 노드 결함이 발생되었을 경우에 빠른 회복을 수행하는 결함허용정보저장 기법을 설계하는 데에 효과적으로 이용될 수 있다.

A Striped Checkpointing Scheme for the Cluster System with the Distributed RAID

Yun Seok Chang†

ABSTRACT

This paper presents a new striped checkpointing scheme for serverless cluster computers, where the local disks are attached to the cluster nodes collectively form a distributed RAID with a single I/O space. Striping enables parallel I/O on the distributed disks and staggering avoids network bottleneck in the distributed RAID. We demonstrate how to reduce the checkpointing overhead and increase the availability by striping and staggering dynamically for communication intensive applications. Linpack HPC Benchamark and MPI programs are applied to these checkpointing schemes for performance evaluation on the 16-nodes cluster system. Benchmark results prove the benefits of the striped checkpointing scheme compare to the existing schemes, and these results are useful to design the efficient checkpointing scheme for fast rollback recovery from any single node failure in a cluster system.

키워드 : 분할된 결함허용정보 저장 기법(Striped Checkpointing Scheme), 결함허용정보(Checkpoint), 클러스터 컴퓨터(Cluster Computer), 분산 RAID(Distributed RAID)

1. 서 론

클러스터 시스템의 각 노드에서 동시에 실행되는 모든 프로세스들은 메시지 전송을 통하여 상호간에 통신을 수행함으로써 동기적인 프로세스 수행이 가능하도록 한다[1]. 그러므로 하나 이상의 노드에서 결함이 발생될 경우에 클러스터 차원에서 전역적인 일관성을 유지하고 최소한의 비용으로 결함 회복(rollback recovery)을 수행하기 위하여, 각 노드들은 자신이 수행하고 있는 프로세스에 대한 정보, 즉 결함허용정보(Checkpoint)를 저장 장치에 주기적으로 저장하여야 한다. 이를 위하여 클러스터 컴퓨터 시스템에 대한 여러 가지 결함 허용 정보 저장 기법들이 연구된 바 있다. 동시적 결함허용정보 저장 기법(Coordinated checkpointing scheme)은 임의의 노드에서 발생할 수 있는 결함에 대하여 전역적인 일관성을 보장하기 위하여 주기적으로 각 노드들의 결함허용정보를 동시적으로 저장 장치에 저장한다[2]. 이 기법은 모든 노드들에 대한 일관성을 유지할 수 있다는 장점을 가지고 있다. 그러나 결함허용정보가 저장되는 동안 프로세스들의 수행이 중단되고, 네트워크 부하가 집중될 뿐만 아니라 입출력 병목 현상이 발생되기도 한다. 이러한 문

※ 이 논문은 2002학년도 대진대학교 학술연구비 지원에 의한 것임.
† 종신회원 : 대진대학교 컴퓨터공학과 교수
논문접수 : 2003년 3월 26일, 심사완료 : 2003년 5월 26일

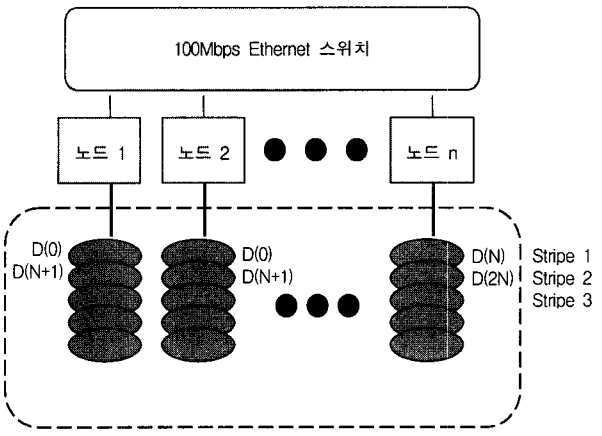
제점을 해결하기 위하여, 디스크를 사용하지 않는 결함허용정보 저장 기법(Diskless checkpointing scheme)이 제안되기도 하였다[3]. 그러나 디스크를 사용하지 않는 결함허용정보 저장 기법은 매우 제한된 결함 복구 성능만을 가지고 있기 때문에 동시 저장 기법이 가진 문제점을 모두 해결하기는 어렵다. 또 다른 방법인 순차적 결함허용정보 저장 기법(Staggered checkpointing scheme)은 각 노드들이 순차적으로 저장 장치에 접근하여 결함허용정보를 저장하므로 입출력 부하를 줄이고 네트워크 병목 현상을 방지할 수 있는 장점을 가지고 있다[4]. 그러나 이 기법은 결함허용정보가 저장되는 노드들 사이에 일관성이 유지되지 않는 문제점을 가지고 있기 때문에 노드간 메시지 기록을 위한 추가적인 오버헤드를 필요로 한다.

동시적 결함허용정보 저장 기법이나 순차적 결함허용정보 저장 기법에서는 파일 서버나 서버에 연결되어 있는 지역 디스크와 같은 집중된 저장 장치를 사용하여 결함허용정보들을 저장한다. 그러나 서버를 사용하지 않는 다중 컴퓨터를 기반으로 하는 클러스터 시스템에서는 각 노드에 연결된 지역 디스크들을 이용하여 분산 RAID를 구현하고 여기에 단일 입출력 공간(SIOS : Single IO Space)의 저장 공간을 구성하여, 결함허용정보를 저장하는 방법이 서버를 이용하는 방법에 비하여 효율적이다[5, 6]. 단일 입출력 공간을 가지는 분산 RAID를 저장 장치로 사용하면 다음과 같은 두 가지 장점을 얻을 수 있다. 첫째로는, 결함허용정보들을 분산 RAID에 병렬 입출력의 장점을 이용하여 저장함으로써 노드들과 저장 장치 사이의 입출력 경쟁을 효과적으로 줄일 수 있다. 두 번째로는, 결함 복구를 위하여 결함허용정보를 여러 디스크로부터 읽어 들이는 데에 따른 지연 시간을 분산 RAID를 이용함으로써 줄일 수 있다. 따라서 클러스터의 평균 복구 시간(MTTR : Mean Time To Repair)을 줄일 수 있고, 이에 따른 클러스터 시스템의 가용성 향상을 기대할 수 있다. 이와 같이 단일 입출력 공간으로 구성된 분산 RAID를 이용하여 결함허용정보를 저장하면, 순차 저장 기법에 동시 저장 기법의 장점들을 부분적으로 결합하여 네트워크 경쟁을 줄이면서 병렬 입출력에 의한 이득을 얻을 수 있는 분할된 결함허용정보 저장 기법(Striped checkpointing scheme)을 구현할 수 있다. 이 기법은 전체 노드들을 여러 개의 분할 집합(Stripe group)으로 분할하고, 한 분할 집합 내에 포함된 노드들에 대한 결함허용정보들을 동시에 분산 RAID에 저장한다. 하나의 분할 집합에 대한 결함허용정보 저장이 완료되면, 다음 분할집합에 포함된 노드들에 대한 결함허용정보들을 저장한다. 이 경우, 각 분할 집합 내에서는 동시적 결함허용정보 저장 기법이 적용되지만, 분할 집합들 사이에는 순차적 결함허용정보 저장 기법이 적용된다. 이와 같은 결함허용정보 저장 기법은 노드간의 입출력 경쟁이 매우 큰 경우에도 분할 집합 내에서 수행하는 병렬 입출력을 통하여 저장 장치 내의 입출력 대기 부하를 줄일 수 있다. 또한 많은 노드로부터의 병렬 입출력으로 인하여 네트워크 부하가 매우 크게 증가될 경우에도 순차적으로 저장 장치를 접근함으로써 네트워크의 병목현상을 방지할 수 있다. 그러므로 클러스터를 구성하는 노드의 수가 적은 경우에는 병렬 입출력의 효과 증대에 중점을 두어야 하고, 노드의 수가 많은 경우에는 네트워크 병목현상을 줄이는 데에 중점을 두어야 한다.

본 연구에서는 이와 같은 분할된 결함허용정보 저장 기법의 성능을 평가하기 위하여 16개의 노드로 구성된 클러스터 컴퓨터 시스템을 구현하고, Linpack HPC(High Performance Computing) 벤치마크와 MPI를 이용하여 기존의 결함허용정보 저장 기법과의 성능을 비교, 분석하였으며 벤치마크 결과를 통하여 분할된 결함허용정보 저장 기법이 동시적 결함허용정보저장 기법이나 순차적 결함허용정보 저장 기법에 비하여 우수한 성능 특성을 가지고 있음을 보인다.

2. 분산 RAID와 단일 입출력 공간

서버를 사용하지 않는 클러스터 시스템에서 각 노드에 연결된 지역 디스크들을 사용하여 분산 RAID를 구성하고, 이를 저장 장치로 사용하기 위해서는 반드시 클러스터에 포함된 모든 디스크들에 대하여 단일 입출력 공간을 구성하여야 한다. 단일 입출력 공간은 높은 확장성과 고가용성, 그리고 입출력 위주의 클러스터 응용들에 대한 호환성을 제공할 수 있어야 하고, 사용자들이 저장 장치의 물리적인 구성이나 디스크 블록의 배치에 대한 이해 없이도 모든 디스크를 이용할 수 있도록 투명성을 제공하여야 한다. (그림 1)은 분산 RAID를 이용한 단일 입출력 공간의 개념을 나타낸다. 여기에서 각 노드에는 지역 디스크가 연결되어 있



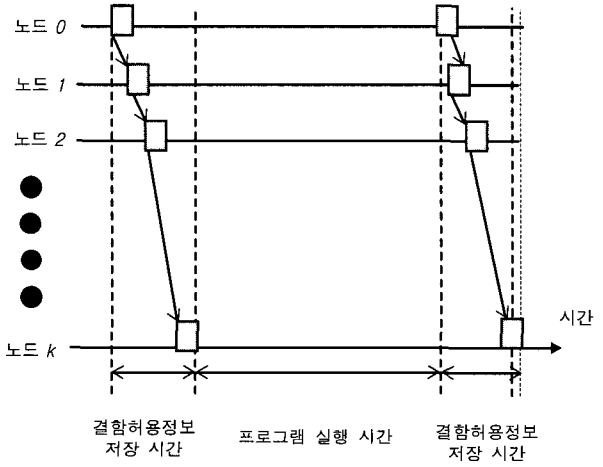
(그림 1) 클러스터 시스템에서의 단일 입출력 공간의 구조

고, 이 지역 디스크들이 100Mbp의 Ethernet 스위치로 구성된 네트워크를 통하여 단일 입출력 공간을 형성하는 분산 RAID를 구성한다. 따라서 파일 서버를 포함하는 클러스터 컴퓨터 시스템은 NFS를 통하여 모든 결함허용정보를 파일 서버에 저장하는 반면에 단일 입출력 공간에서는, 결함허용정보들이 분산 디스크들 전체에 걸쳐서 저장된다.

분산 RAID는 확장성 있는 클러스터 시스템을 구축하는데 있어서 필수 불가결한 요소이다[7]. 그리고 단일 입출력 공간은 시스템 커널에서 동작하는 장치 드라이버들간의 결합에 의하여 구현된다. 이 장치 드라이버들은 물리적으로 분산된 모든 디스크들에 걸쳐서 단일 입출력 공간을 구성하기 위하여 상호작용을 수행하면서 동작한다. 이를 통하여 클러스터 시스템은 서버 없이 공통의 저장 장치를 구성할 수 있으며, 커널 수준에서 다른 노드에 연결된 디스크에 대하여 직접적으로 접근할 수 있도록 할 수 있다. 클러스터 내에 있는 모든 분산된 디스크들이 단일 입출력 공간을 형성할 경우, 지역 디스크들의 어떤 분할 집합에서도 병렬 입출력이 가능하며, 많은 시스템 호출을 발생시키지 않고도 다른 노드에 있는 파일을 접근할 수 있는 특성을 제공한다.

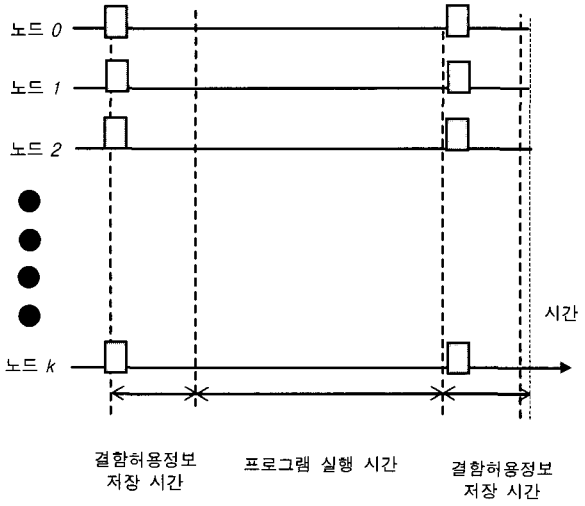
3. 분할된 결함허용정보 저장 기법

순차적 결함허용정보 저장 기법의 개념은 클러스터 시스템 전체에 걸쳐서 각 노드들이 결함허용정보를 순차적으로 저장 장치에 저장할 수 있도록 하는 것이다. (그림 2)는 순차적 결함허용정보 저장 기법의 결함허용정보 저장 과정을 나타내고 있다. 각 노드의 결함허용정보들은 일정한 프로그램 실행 주기마다 저장 장치에 저장되고, 저장 장치는 한 번에 한 노드만이 접근할 수 있다. 따라서 한 노드가 결함허용정보 저장을 완료한 후에 다음 노드가 결함허용정보를 저장 장치에 저장하도록 되어 있다. 비록 순차적 결함허용정보 저장 기법이 결함허용정보를 저장할 때 프로세스들 사이에서 발생될 수 있는 입출력 충돌을 방지하고, 입출력 부하와 네트워크 부담을 경감시키는 장점을 가지고 있지만, 앞서 설명한 바와 같이 메시지 기록에 드는 오버헤드가 크고, 각 노드의 결함허용정보들 사이에 일관성을 유지하기 위하여 많은 메시지 제어 프로세스를 필요로 한다[8]. 따라서 순차적 결함허용정보 저장 기법은 클러스터의 크기가 커질수록 결함허용정보 저장 비용이 급격히 증가하게 되며, 메시지 기록을 저장하기 위한 데이터의 용량과 요구되는 네트워크 대역폭이 증가된다.



(그림 2) 순차 저장 기법에서의 결함허용정보 저장

동시적 결함허용정보 저장 기법은 모든 노드들에서 수행되는 프로세스들에 대한 결함 허용 정보를 동시에 저장 장치에 저장한다(그림 3). 따라서 동시적 결함허용정보 저장 기법은 넓은 네트워크 및 입출력 대역폭을 요구하게 되고, 순차적 결함허용정보 저장 기법과 마찬가지로 클러스터의 크기가 증가되면 여러 프로세스들이 동시에 쓰기 동작을 수행함으로써 심한 네트워크 경쟁을 유발하고, 입출력 병목 현상을 일으키게 된다.

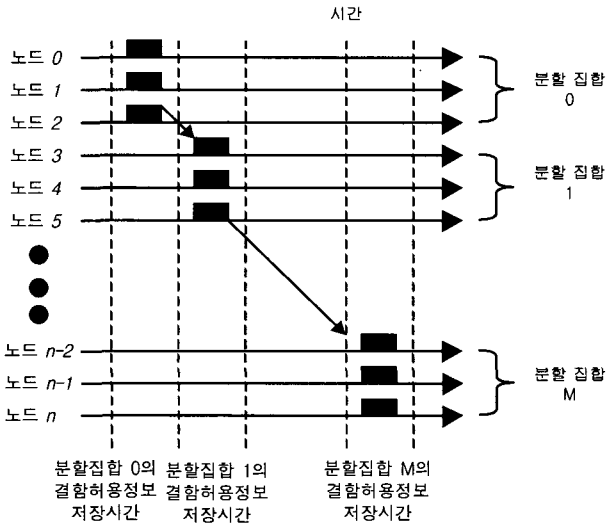


(그림 3) 동시저장기법에서의 결함허용정보 저장

분산 RAID의 병렬 입출력 능력은 클러스터 시스템에서 결함허용정보를 빠르게 저장하는 데에 효과적으로 이용될 수 있다. 분할된 결함허용정보 저장 기법에서는 결함허용정보들을 분할 집합을 형성하고 있는 디스크들에 분산 저장할 수 있다. 하나의 분할 집합은 병렬적으로 접근할 수 있는 디스크들의 부분집합으로 정의된다. 동일한 분할 집합 내에 포함된 디스크들은 결함허용정보 저장을 위한 입출력 동작을 병렬로 동시에 실행할 수 있다. 또한 네트워크의 부하를 줄이기 위해서, 각 분할 집합들은 결함허용정보 저장을 위한 입출력 동작을 순차적으로 수행한다. 따라서 각 분할 집합들에 대한 입출력은 병렬 입출력을 수행하

면서도 순차 입출력을 이용하여 입출력 경쟁을 줄일 수 있게 된다.

분할된 결함허용정보 저장 기법에서 입출력의 병렬성을 최대로 이용할 경우에는 분할 집합의 크기가 클러스터의 크기가 되며, 이 경우는 동시적 결함허용정보 저장 기법과 동일한 구조가 된다. 또한 분할 집합의 크기를 최소, 즉 1로 하면 순차적 결함허용정보 저장 기법과 동일한 구조가 된다. 따라서 분할된 결함허용정보 저장 기법은 병렬 입출력과 순차 입출력의 장점을 병용한 기법이라고 할 수 있다. (그림 4)는 이와 같은 분할된 결함허용정보 저장 기법의 동작 개념을 나타내고 있다.



(그림 4) 분할 저장 기법에서의 결함허용정보 저장 방법

이와 같은 분할된 결함허용정보 저장 기법은 네트워크의 이용률과 입출력 성능을 모두 향상시킬 수 있다. 여기에서, 분할 집합의 크기와 분할 집합의 수는 가변적으로 결정될 수 있다. 특정한 응용 프로그램에 대하여, 사용자는 상대적인 분할 집합의 크기와 수를 가장 효과적인 결함허용 복구 성능을 나타낼 수 있도록 조정할 수 있다. 분할 집합의 크기가 클수록 병렬 입출력의 크기를 증가시켜서 디스크 대역폭을 효율적으로 사용할 수 있고, 분할 집합의 수를 증가시킬수록 네트워크 경쟁을 줄일 수 있게 된다.

4. 성능 평가 및 분석

4.1 벤치마크 환경

분할된 결함허용정보 저장 기법과 기존의 결함허용정보 저장 기법들의 성능을 비교 평가하기 위하여 본 연구에서는 MPI(Message Passing Interface)를 이용하여 노드간 메시지 전송을 수행하는 Linpack HPC(High-Performance Computing) 벤치마크를 사용하였다. 본 연구에서는 Libckpt[9] 라이브러리를 이용한 프로그램 함수를 HPC 벤치마크 프로그램에 첨가함으로써 시스템에 추가적인 부하를 가하지 않고 결함허용정보를 저장하는 알고리즘을 구현하여 Linux 클러스터 환경에서 동작될 수 있는 벤치마크 프로그램을 구성하였다. 클러스터 시스템의 결함허용 성능은 메시지 기록에 소요되는 시간을 포함하여 결함허용정보를 저장하는 데 걸리는 시간과 입출력 처리량을 통하여 직접적으로 나타낼 수 있다.

벤치마크를 실행시키기 위하 클러스터 환경은 <표 1>과 같다. 클러스터를 구성하는 노드로는 16대의 PC를 사용하였으며 100Mbps의 Ethernet 전용 스위치를 통하여 상호간에 연결된다. 각 노드는 RedHat Linux 6.2를 운영체제로 사용하고, 128MB의 메인 메모리와 40GB의 용량을 가지는 하나의 하드디스크를 포함하고 있으며 각 디스크 공간 중 4GB 영역을 클러스터 저장 장치 공간으로 할당하였다.

〈표 1〉 클러스터 노드 구성

구성 요소	규 격
CPU	Intel Pentium-II 233MHz
메인메모리 크기	128Mbyte SDRAM
L2캐쉬 크기	256Kbyte
디스크	40GB Seagate 4GB for SIOS, 36GB for OS
네트워크	100Mbps Intel Express Pro100

결함허용정보를 저장하는 저장 구조는 네트워크 파일 시스템(NFS) 구조와 단일 입출력 공간(SIOS) 구조의 두 가지로 구성하였다. NFS 구조에서는 지정된 노드에 연결된 하나의 디스크에만 결함허용정보들을 저장한다. 반면에 SIOS 구조에서는 각 노드의 결함허용정보들이 분산 RAID를 기반으로 하는 단일 입출력 공간에 저장된다. 따라서 NFS에서는 디스크 블록주소는 하나의 디스크에만 매핑되고, 단일 입출력 공간에서는 디스크 블록주소들이 각 디스크들에 분산되어 매핑이 이루어지게 된다. 이 경우, 단일 입출력 공간을 이루기 위하여 최소 2개에서 최대 16개까지의 디스크들이 분산 RAID에 참여할 수 있지만, 본 실험에서는 16개 노드 중 4개, 또는 8개의 노드에 연결된 디스크들을 사용하여 단일 입출력 공간을 구성하였다.

벤치마크를 통한 성능 평가에서는 동시 저장 기법과 순차 저장 기법, 그리고 본 논문에서 제안한 분할 저장 기법의 3가지 결함허용정보 저장 기법이 적용되었다. 동시 저장 기법에서는 주기적으로 모든 노드들이 동시에 결함허용정보를 저장 장치에 저장하고, 모든 노드의 결함허용정보 저장 작업이 종료되면 다음 처리를 수행하도록 한다. 비록 본 연구에서 이용된 클러스터 시스템이 16개의 노드만을 사용함으로써 동시 저장 기법의 결정적인 문제점인 네트워크

병목현상이 두드러지게 나타나지는 않지만, 순차 저장 기법에 대한 상대적인 성능을 나타는 데에는 충분한 크기를 가지고 있다. 순차 저장 기법에서는 주기적으로 결함허용정보들을 저장할 때 한 번에 한 노드씩 저장 장치에 저장하도록 하였다. 분할 저장 기법에서는 한 분할 그룹에 2개, 4개, 또는 8개의 노드를 포함하는 3가지 분할 그룹 크기를 사용하였다.

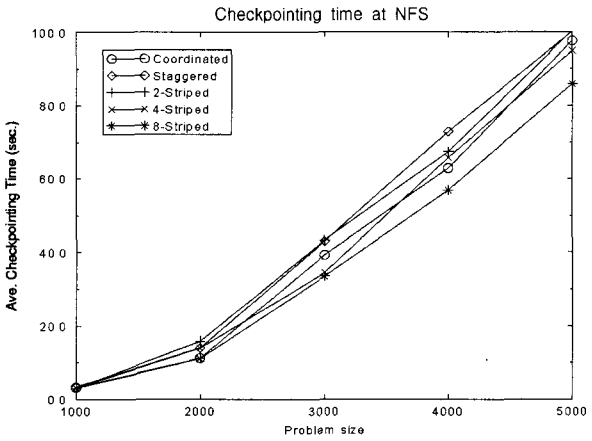
성능 평가를 위하여 사용된 Linpack HPC 벤치마크에는 100×100 크기와 1000×1000 크기의 행렬 연산을 수행하는 일련의 문제들이 포함되어 있으며 Linpack의 핵심 내용인 LAPACK과 BLACS 라이브러리의 MPI 버전이 포함되어 있다. <표 2>는 HPC 벤치마크 프로그램을 본 논문에서 구현한 16 노드의 클러스터 시스템에서 실행시킨 예를 보이고 있다. 여기서는 시스템 부하를 문제크기로 변화시킬 수 있으며 지수 P와 Q를 각각 4로 할당하여 문제크기 N을 1000에서 10000까지 부가하였을 때 클러스터가 나타내는 처리 성능을 평가하였다.

〈표 2〉 16 노드 클러스터에서 HPC 벤치마크 수행 예

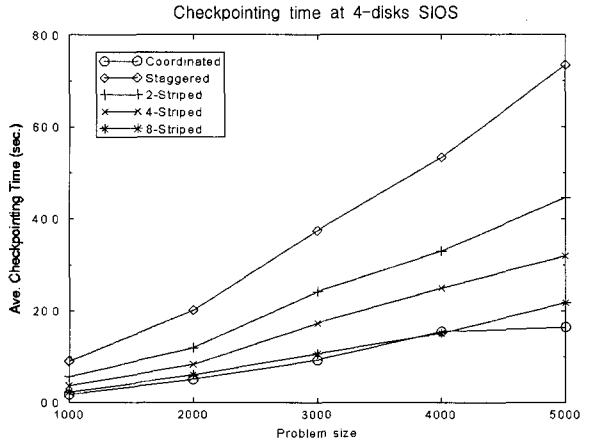
문제크기 N	NB	P값	Q값	결함허용정보 크기(MB)	실행시간	처리성능 Gflops
1000	60	4	4	2.3	1.73	3.861e-01
1000	60	4	4	2.3	1.71	3.901e-01
1500	60	4	4	5.8	3.74	6.027e-01
1500	60	4	4	5.8	3.86	5.830e-01
2000	60	4	4	9.3	6.79	7.863e-01
2000	60	4	4	9.3	7.02	7.606e-01
2500	60	4	4	13.4	12.12	8.601e-01
3000	60	4	4	13.4	20.45	8.808e-01
4000	60	4	4	18.2	46.64	9.153e-01
5000	60	4	4	18.2	96.55	8.635e-01
6000	60	4	4	25.1	183.58	7.847e-01
7000	60	4	4	25.1	290.35	7.878e-01
8000	60	4	4	36.4	443.87	7.692e-01
9000	60	4	4	36.4	634.74	7.659e-01
10000	60	4	4	49.7	871.54	7.651e-01

본 연구에서 구현한 클러스터 시스템은 최대성능점(N = 4000)에서 0.91Gflops의 성능을 나타내며 부하가 증가함에 따라서 성능이 증가되다가 최대성능점을 경과하면 성능이 차츰 저하된다. 클러스터 노드의 성능과 클러스터의 크기가 증가할수록 높은 N 값에서 최대성능점이 형성된다. 본 논문에서는 성능 평가에 부가하는 부하의 크기를 최저성능에서 최대성능 사이에서 부가하도록 하였다. 따라서 부하의 크기는 1000에서 5000까지 가변적으로 부가되었다. 본 성능 평가에서는 이들 라이브러리 프로그램에 Libckpt 함수를 결합하여 주기적으로 결함허용정보를 저장하도록 프로그램을 수정하였다. 결함허용정보의 크기는 부가되는 부하의 크기에 비례하여 증가되기 때문에 부하가 증가될수록 입출력 병렬성의 효과는 증대된다.

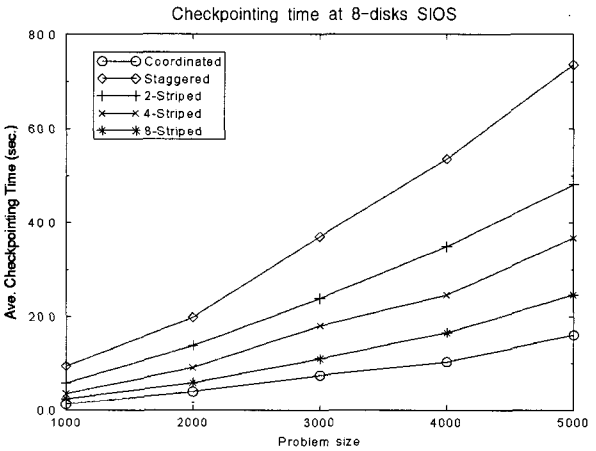
4.2 성능 평가 분석



(그림 5) NFS에서의 결함허용정보 저장 시간



(a) 4개의 디스크로 SIOS가 구성된 경우



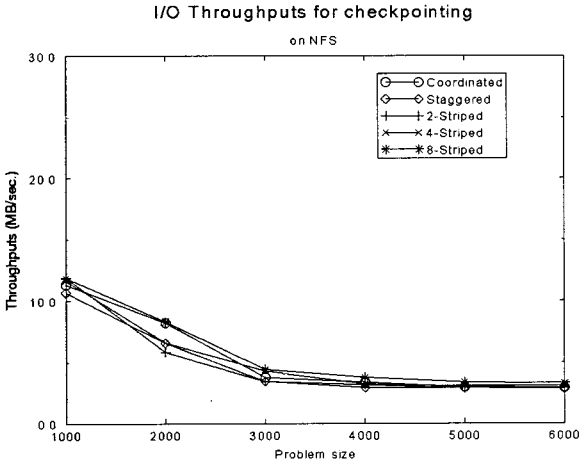
(b) 8개의 디스크로 SIOS가 구성된 경우

(그림 6) SIOS에서의 결함허용정보 저장 시간

(그림 5)와 (그림 6)은 부하의 크기가 변화할 때에 각 결함허용정보 저장 기법들이 NFS, 4개의 디스크를 사용한 SIOS, 그리고 8개의 디스크를 사용한 SIOS 구조의 저장 장치에

각각 결함허용정보를 저장하는데 걸리는 평균 결함허용정보 저장 시간을 나타낸다. 여기서 평균 결함허용정보 저장 시간은 결함허용정보를 디스크에 저장하는 시간, 메시지 기록에 소요되는 시간, 그리고 결함허용정보들을 저장하는 동안에 소요된 노드간 MPI 통신 시간들이 모두 포함된 전체 입출력 시간을 나타낸다.

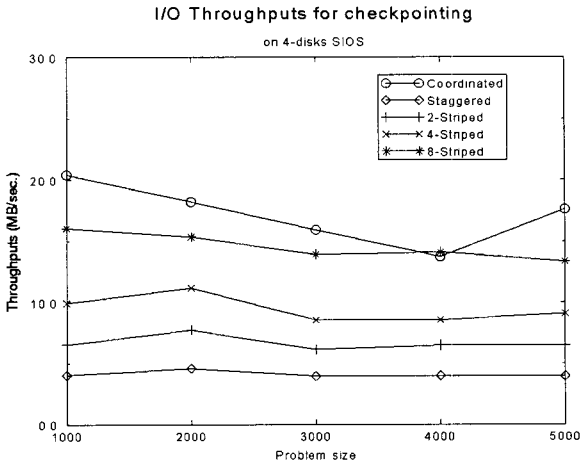
또한 (그림 7)과 (그림 8)은 NFS와 4개, 또는 8개의 디스크들을 사용한 SIOS 저장 구조에서 각 결함허용정보 저장 기법들의 입출력 처리량을 나타낸다. 입출력 처리량은 결함허용정보들이 저장 장치에 저장되는 동안 수행되는 입출력을 처리하는 능력으로, 각 노드로부터 네트워크 파일시스템이나 분산 RAID로 결함허용정보들이 전송되는 속도를 나타낸다.



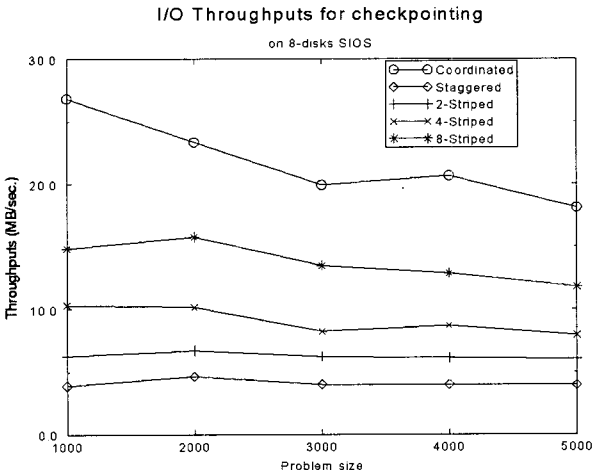
(그림 7) NFS에서의 입출력 처리량

실험 결과에서는 평균 결함허용정보 저장 시간과 입출력 처리량에 있어서 동시적 결함허용정보 저장 기법이 가장 우수한 성능을 보여주고 있다. 이는 본 실험에서 사용한 클러스터의 크기, 즉 클러스터 노드수가 네트워크의 대역폭에 비하여 작기 때문에 네트워크 병목 현상이 거의 발생되지 않는다는 사실을 보여주고 있다. 그러나 기존의 연구들에서 클러스터의 크기가 증가하여 네트워크를 통하여 전송되는 데이터가 전송 용량을 초과하게 되면 심각한 네트워크 경쟁과 병목현상을 발생시킨다는 결과를 보여 주고 있다. 앞 절에서 설명한 바와 같이 본 연구에서는 비교적 적은 수의 노드를 사용하는 클러스터 시스템을 사용하였기 때문에 분할된 결함허용정보 저장 기법은 동시적 결함허용정보 저장 기법보다는 순차적 결함허용정보 저장 기법과의 비교에서 특성과 성능의 차이가 나타나고 있다.

NFS 구조를 저장 장치로 사용하는 경우에는 결함허용정보 저장 시간과 입출력 처리량에 대하여 각 결함허용정보 저장 기법들 사이에 큰 성능 차이를 보이지 않는다. 이는 저장 장치에 대한 입출력 부하가 네트워크 부하에 비하여



(a) 4개의 디스크로 SIOS를 구성한 경우



(b) 8개의 디스크로 SIOS를 구성한 경우

(그림 8) SIOS에서의 입출력 처리량

큰 비중을 차지하고 있기 때문인 것으로 분석할 수 있다. 본 연구에서 수행한 3가지 결함허용정보 저장 기법은 입출력 부하보다는 네트워크 부하를 경감시키는 것을 목적으로 한다. 따라서 NFS 구조를 저장 장치로 사용하는 클러스터 시스템에서는 입출력 부하가 증가될수록, 결함허용정보 저장 기법의 차이는 성능에 큰 영향을 미치지 않는다.

반면에 SIOS 구조를 저장 장치로 사용하는 경우에는 분산 RAID의 병렬 입출력 효과가 크게 작용하는 것을 볼 수 있다. SIOS에서, 병렬 입출력을 이용하는 동시적 결함허용정보 저장 기법과 분할된 결함허용정보 저장 기법이 순차적 결함허용정보 저장 기법에 비하여 우수한 성능을 보인다. 또한 분할된 결함허용정보 저장 기법에서는 하나의 분할 집합이 사용하는 병렬 입출력 대역폭이 전체 네트워크 용량을 초과하지 않는 범위 내에서는 분할 집합의 크기가 클수록 우수한 성능을 나타낸다. 본 연구에서 구현한 클러스터 시스템에서, 분할 집합의 크기가 8인 경우에는 순차적

결함허용정보 저장 기법에 비하여 결함허용정보 저장 시간을 1/4 이하로 줄일 수 있고 입출력 처리량은 3배 이상 증대시킬 수 있음을 보이고 있다. Linpack HPC 벤치마크에서는 부하가 증가할수록 저장하여야 할 결함허용정보의 크기가 증가되기 때문에 부하가 증가함에 따라서 순차적 결함허용정보 저장 기법과 분할된 결함허용정보 저장 기법간의 성능 차이가 증가하는 사실을 알 수 있다. 따라서 부하가 큰 프로세스를 수행하는 클러스터 시스템일수록 분할된 결함허용저보 저장 기법이 순차적 결함허용정보 저장 기법에 비하여 우수한 결함허용정보 저장 성능을 보이게 된다.

입출력 처리량의 측면에서는 분할된 결함허용정보 저장 기법이 순차적 결함허용정보 저장 기법에 비하여 역시 우수한 성능을 보인다. 그러나 부하가 매우 커지면 분할된 결함허용정보 저장 기법의 입출력 처리 성능이 조금씩 저하된다. 이는 부하가 증가함에 따라서 분산 RAID에서의 입출력 병목 현상이 점차로 발생되기 때문이며, 이는 동시적 결함허용정보 저장 기법에 가장 많은 영향을 끼친다. 그러나 높은 부하에서도 순차적 결함허용정보 저장 기법에 비하여 분할된 결함허용정보 저장 기법이 여전히 우수한 성능을 보이고 있다.

이러한 결과들은 네트워크 용량 한도 내에서 최대 크기의 분할 집합으로 구성된 분할된 결함허용저보저장 기법이 분산 RAID를 기반으로 하는 결함허용정보 저장에 있어서 가장 우수한 성능을 나타낼 수 있음을 보여주고 있다. 클러스터의 크기가 증가될 경우, 동시적 결함허용정보 저장 기법이 적용된다면 네트워크 용량을 초과할 정도로 많은 입출력 부하를 발생시키게 되지만, 적절한 분할 집합 크기를 가지는 분할된 결함허용정보 저장 기법을 적용하면 네트워크 대역폭을 효율적으로 이용하면서 우수한 결함허용정보 저장 성능을 나타낼 수 있게 된다. 이는 분할된 결함허용정보 저장 기법에서는 수행하여야 할 프로세스가 발생시키는 부하의 크기와 클러스터의 크기에 따라서 네트워크의 대역폭과 분할 집합의 크기 사이에 적절한 조정이 필요하다는 것을 보여 준다. 그러므로 결함허용정보 저장을 통한 결함 복구 능력을 가지는 클러스터 시스템을 설계할 때, 분할된 결함허용정보 저장 기법을 적용하고, 적절한 크기의 분할 집합을 설정하도록 하면, 결함허용정보 저장 성능을 효과적으로 높일 수 있다.

5. 결론 및 향후 연구

본 연구에서 수행한 성능 평가 결과, 분산 RAID를 저장 장치로 사용하는 클러스터 시스템에서의 결함허용정보 저장 기법은 분할된 결함허용정보 저장 기법이 가장 효과적이라고 할 수 있다. 분할 집합에 의한 병렬 입출력 동작은 동시에 여러 디스크에 대한 접근을 가능하게 하고, 저장 장치에 대한 순차적인 접근은 각 디스크에서의 입출력 병목 현상을 줄일 수 있다. 따라서 클러스터의 크기가 증가되어, 입출력 대역폭과 네트워크 용량을 초과하는 경우에, 분할된 결함허용정보 저장 기법은 다른 결함허용정보 저장 기법에 비하여 우수한 성능을 보인다.

분할된 결함허용정보 저장 기법에서 분할 집합의 크기는 서로 다른 부하를 발생시키는 여러 응용에 대하여 동적으로 조정할 수 있도록 하여야 한다. 이러한 동적인 조정을 통하여 주어진 응용에 대하여 가장 최적의 분할 집합을 사용하는 결함허용정보 저장 기법을 적용할 수 있게 된다. 병렬성을 많이 이용할수록 디스크에 대한 입출력 대역폭을 확대시킬 수 있다. 또한 분할 집합의 수가 많을수록 네트워크 경쟁 문제를 용이하게 해결할 수 있다. 그러므로 분산 RAID를 기반으로 하는 결함허용정보 저장 기법을 사용한 결함 복구 성능이 특정 응용에 대하여 가장 효과적으로 적용될 수 있도록 하기 위해서는 분할 집합의 크기와 수가 적절하게 결정되어야 한다. 그러나 본 연구는 충분한 크기의 클러스터 시스템을 사용하지 못하였기 때문에 동시적 결함허용정보 저장 기법에 대한 성능상의 차이점을 명확하게 비교하기 어려운 문제점을 가지고 있다. 따라서 차후에는 32노드 이상의 대형 클러스터 시스템을 구현하여 동시적 결함허용정보 저장 기법에 대한 분할된 결함허용정보 저장 기법의 성능 차이를 분석하고 응용의 부하에 따라서 분할 집합의 크기를 동적으로 결정할 수 있는 결함허용정보 저장 기법을 설계하여 기존의 결함허용정보 저장 기법에 대한 성능 평가를 수행하도록 할 예정이다.

참 고 문 헌

[1] K. Hwang and Z. Xu, "Scalable Parallel Computing," McGraw-Hill, 2000.

[2] G. Cao and M. Singhal, "On Coordinated Checkpointing in Distributed Systems," *IEEE Transactions on Parallel and Distributed Systems*, Vol.9, No.12, 1998.

[3] J. Plank K. Li and M. Puening, "Diskless Checkpointing," *IEEE Transactions on parallel and Distributed Systems*, 1998.

[4] N. Vaidya, "Staggered Consistent Checkpointing," *IEEE Transactions on Parallel and Distributed Systems*, Vol.10, No.7, 1999.

[5] K. Hwang, H. Jin, R. Ho and W. Ro, "Reliable Cluster Computing with a New Checkpointing RAID-x Architecture," *Proceedings of 9-th Workshop on Heterogeneous Computing*, Cancum, Mexico, 2000.

[6] K Hwang, H. Jin and R. Ho, "RAID-x : A New Distributed

Disk Array for I/O-Centric Cluster Computing," *Proceedings of 9th High-Performance Distributed Computing Symposium*, Pittsburgh, 2000.

[7] K. Hwang, H. Jin, E. Chow, C. Wang and Z. Xu, "Designing SSI Clusters with Hierarchical Checkpointing and Single IO Space," *IEEE Concurrency Magazine*, 1999.

[8] E. Elnozahy and W. Zwaenepoel, "On the Use and Implementation of Message Logging," *Proceedings of 24th International Symposium on Fault-Tolerant Computing*, 1994.

[9] J. Plank, M. Beck, G. Kingsley and K. Li, "Libckpt : Transparent Checkpointing Under UNIX," *Proceedings of USENIX Winter 1995 Technical Conference*, 1995.

장 윤 석

e-mail : cosmos@daejin.ac.kr

1988년 서울대학교 물리학과(이학사)

1990년 서울대학교 대학원 컴퓨터공학과 (공학석사)

1998년 서울대학교 대학원 컴퓨터공학과 (공학박사)

1994년~2002년 대진대학교 컴퓨터공학과 전임강사 및 조교수

2000년~2001년 Visiting Scholar in Dept. of EE-Systems, University of Southern California

2003년~현재 대진대학교 컴퓨터공학과 부교수

관심분야 : 컴퓨터구조, 클러스터 컴퓨터, RAID, 전자상거래 구조 등