제26회 한국정보처리학회 추계학술발표대회 논문집 제13권 제2호 (2006. 11)

쓰레드를 이용한 대규모 파일 관리 시스템 구현

김성준*, 우준, 최윤근, 김중권
*한국과학기술정보연구원 슈퍼컴퓨팅사업팀
e-mail:sjkim@kisti.re.kr

A Implementation of management system for a large temporary file using thread

Sung-Jun Kim*, Joon Woo, Youn-Keun Choi, Joong-Kwon Kim
*Korea Institute of Science and Technology Information

요 약

한국과학기술정보연구원에서는 대규모 파일시스템을 IBM p690시스템에서 운영 중에 있다. 이 파일시스템은 정해진 쿼터 정책에 따라서 운영되는 홈 파일시스템과 사용자의 작업 수행시에 필요로 하는 입력, 출력파일 및 임시 파일을 저장하는 스크래치 파일시스템으로 사용되고 있다. 공용으로 사용되어지는 스크래치 파일시스템에 일정 기간이 경과한 파일들이 무분별하게 남아있게 되면 디스크의 가용 공간을 차지하게 되어 정상적인 작업수행에 방해를 받게 된다. 본 고에서는 스크래치 파일시스템에 존재하는 파일들을 검사하여 일정 기간이 지나서도 남아 있는 사용자 파일을 삭제함으로써 스크래치 파일시스템의 가용 공간을 확보하는 프로그램을 개발하였다.

1. 서론

한국과학기술정보연구원에서는 IBM p690시스템에서 약 30TB 대용량 파일시스템을 운영 중에 있으며, 향후 이보다 더 대규모의 파일시스템을 사용하는 슈퍼컴퓨터 4호기를 도입 예정에 있다.

이 대용량 파일시스템은 사용자의 공유 홈 디렉토리와 작업을 위한 스크래치 디렉토리로 사용되고 있다. 홈 디렉토리의 경우 사용자마다 정해진 쿼터 정책에 따라 디스크를 할당받아서 사용되며, 스크래치 디렉토리는 사용자들의 작업이 수행되는 과정에서 필요한 입력·결과파일 및 임시 작업파일을 저장하는 목적으로 공용으로 사용되고 있다.

공용으로 사용되어지는 스크래치 디렉토리에 일정 기간이 경과한 파일들이 무분별하게 남아있게 되면 디스크의 가용공간을 잠식하게 되어 가용 디스크 공간이 없어질 수도 있다. 이런 경우 수행중인 작업의 비정상 종료를 야기할 수도 있기 때문에 이를 방지하기 위해서 기존에 스크립트 언어를 이용하여 주기적으로 일정 기간 동안 사용하지 않은 파일을 삭제하는 프로그램을 개발하여 적용하여 왔다.

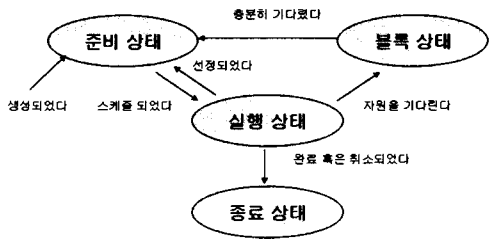
기존의 프로그램을 이용하여 스크래치 디렉토리의 파일들을 검색하고 일정 기간 동안 사용되지 않은 파일을 삭제하는데 8~9시간 정도 소요된다. 향후 기존의 파일시스템 규모보다 수십배 큰 페타바이트급의 파일시스템이 도입될 슈퍼컴퓨터 4호기에 이를 적용할 경우 수십 시간이 소요될 수 있다.

이에 본 고에서는 기존의 스크래치 파일 관리 프로그램의 수행 시간을 단축하기 위해서 포직스 쓰레드(POSIX Thread)를 이용한 병렬처리가 가능한 파일 삭제 프로그램을 설계 및 구현하였다.

2. 관련연구

2.1 POSIX 쓰레드(Pthread) 개요

Pthread는 POSIX Thread로 IEEE에서 표준(POSIX 1003.1c-1995)으로 규격화한 쓰레드 라이브러리이며, User Space에서 작동하는 쓰레드 라이브러리이다. 즉 쓰레드 관리를 위해서 Kernel이 관여를 하지 않고 쓰레드 관련 라이브러리에서 대부분의 쓰레드 관리를 처리하게 된다.



(그림 1) 쓰레드 상태들간의 변화

2.2 Pthread 종류

2.2.1 Kernel Space 쓰레드

커널에서 직접 쓰레드를 관리하며, 커널자체에서 쓰레드 정보를 가진다. 또한 각 프로세스 간의 스케쥴 역시 커널에서 맡는다. 이 경우 I/O 봉쇄 같은 문제는 신경 쓰지 않아도 되며, 또한 SMPs 자원을 제대로 활용할 수 있다는 장점을 가진다

2.2.2 User Space 쓰레드

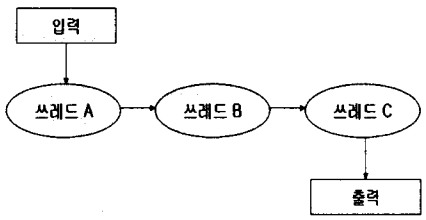
User Space 쓰레드는 쓰레드를 위지하기 위해서 필요한 프로그램 코드, 데이터, 스택, Signal 테이블과 같은 정보를 커널에서 관리하도록 하지 않고 쓰레드 라이브러리에서 직접 처리하게 된다. 또한 각 쓰레드간 협력이 가능하도록 switch 기능을 제공하는데, 커널쓰레드보다 일반적으로 빠른 switch 기능을 제공한다.

2.3 쓰레드 프로그래밍 모델

쓰레드 프로그래밍 모델은 무수한 변종이 존재할 수 있지만 기본적인 모델은 다음 세 가지 모델이다.[1]

2.3.1 파이프라인(pipeline)

각 쓰레드는 데이터 집합 열에 대해 동일한 작업을 반복적으로 수행하며, 그 결과물을 다음 단계의 쓰레드로 넘겨준다. 이 모델은 조립라인이라고도 불리기도 한다.

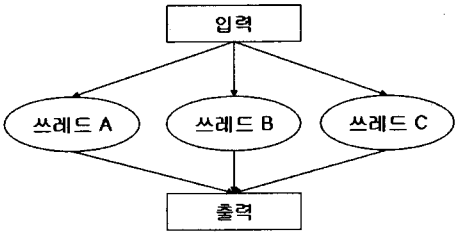


(그림 2) 파이프라인 모델

예를 들어, 스캔된 이미지에 대해 어떤 작업을 수행하는 경우, 쓰레드 A는 이미지 배열을 처리하고, 쓰레드B는 처리된 데이터에서 특정한 특징을 지니고 있는 영역을 검색한다. 마지막으로 쓰레드C는 쓰레드B의 검색결과를 차례로 수집하여 보고서로 출력을 한다.

2.3.2 작업 집단(work crew)

각 쓰레드는 그 자신의 데이터에 대해 작업을 수해한다. 한 작업 집단 내의 쓰레드들은 모두 동일한 작업을 수행할 수도 있으며, 각 쓰레드가 분리된 작업을 수행할 수도 있다. 그러나 한 집단 내의 쓰레드들은 항상 독립적으로 수행된다.

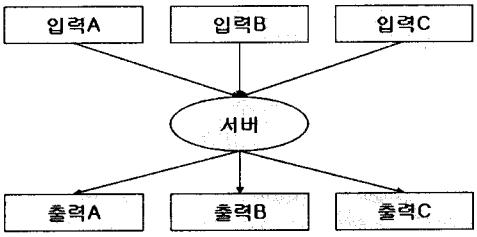


(그림 3) 작업 집단 모델

예를 들면, 쓰레드 집합을 생성하여 각 쓰레드가 배열의 행이나 열의 일부를 처리하도록 할 수 있다. 하나의 데이터 집합은 여러 쓰레드들 간에 나누어 처리되며, 그 결과는 단일 데이터 집합이 된다. 이런 모델을 종종 SIMD(Single Instruction, Multiple Data) 병렬 처리라고 불리기도 한다.

2.3.3 클라이언트 서버(client/server)

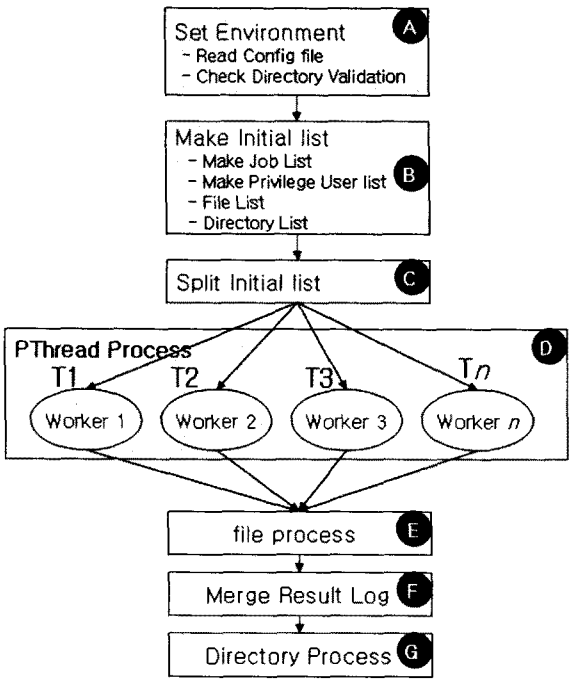
클라이언트는 독립적인 서버와 각 작업을 수행하기 위한 일종의 계약을 맺는다. 종종 계약은 익명의 형태로 이루어지기도 한다. 즉 작업 아이템을 큐에 삽입하는 인터페이스를 통해 요청을 수행할 수 있다.



(그림 4) 클라이언트/서버 모델

3. 설계 및 구현

본 고에서 구현할 프로그램은 쓰레드간의 간섭을 최소화하기 위해서 입력 파일을 각 쓰레드 별로 분배하였으며, 출력 또한 각 쓰레드 별로 기록하도록 하였다. 이는 앞서 설명한 작업 그룹 쓰레드 프로그래밍 모델을 기반으로 한다. 다음 (그림 5)는 전체 프로그램의 개략적인 순서도 이다.



(그림 5) Flow Chart

삭제 대상이 되는 파일들은 다음의 조건을 만족해야 한다.

* 마지막 접근일이 기준일 이전인 파일들
* 현재 수행중인 작업이 없는 사용자들의 파일들
* 미리 정의된 예외 사용자의 파일들은 제외

3.1 모듈 구성

3.1.1 환경 설정 모듈

프로그램의 실행에 앞서 각종 환경 변수를 읽어들이는 모듈이다. 정의되는 환경변수는 생성할 쓰레드의 개수, 처리 대상이 되는 파일의 기준 날짜, 실행 디렉토리, 로그 디렉토리, 작업관리 프로그램 종류 등이다. 디렉토리의 경우에는 실제 존재하는지, 접근 가능한지 디렉토리의 유효성을 검사한다.

3.1.2 초기 파일 리스트 생성 모듈

삭제 대상 파일의 검사에 사용되는 각종 리스트를 생성하는 모듈이다. 현재 실행중인 작업 목록, 예외 사용자 목록, 대상 파일시스템의 상위 디렉토리에 존재하는 일반 파일들의 목록 및 하위 디렉토리들의 목록을 생성한다.

이때 생성되는 파일 목록 및 디렉토리 목록은 예외 사용자들의 파일 및 디렉토리들은 제외되어서 생성된다.

3.1.3 작업 목록 생성 모듈

이전 단계에서 생성한 디렉토리 목록을 생성할 쓰레드의 개수에 맞추어 분할된 입력파일을 생성하는 모듈이다. 쓰레드를 이용한 처리는 각 쓰레드 별로 각자의 입력 파일을 읽어서 처리를 하게 함으로써 각 쓰레드 간의 간섭을 최소화하도록 설계하였다. 이전 단계에서 생성된 파일 목록과 디렉토리 목록 중에서 파일목록은 처리 대상이 되는 파일의 개수가 적기 때문에 순차처리로 진행하며 처리 대상이 많은 디렉토리 목록을 쓰레드를 이용하여 처리하기 위해서 전체 디렉토리 목록을 생성할 쓰레드의 개수에 맞게 분할한다.

3.1.4 쓰레드 처리 모듈

작업 목록 생성 모듈에서 생성된 분할된 디렉토리 목록을 쓰레드별로 처리하는 모듈이다. 디렉토리내에 포함된 파일들을 대상으로 마지막 접근일(last modification date), 파일 소유자의 작성 수행 유무를 검사한다. 이때, 일반 파일(Regular file)이 아닌 링크 파일, 특수 파일, 하위 디렉토리 등은 처리 대상에서 제외하였으며 처리한 결과는 각 쓰레드별 로그 파일에 기록된다.

3.1.5 파일 목록 처리 모듈

작업 대상 파일시스템의 루트 디렉토리에 존재하는 파일들을 처리하는 모듈이다. 디렉토리를 기준으로 처리를 하기 때문에 루트 디렉토리에 존재하는 파일들에 대하여는 별도의 처리를 하여야만 한다. 디렉토리 처리와 마찬가지로 마지막 접근일, 소유자의 작업 수행 유무 등을 검사한다.

3.1.5 로그 취합 모듈

쓰레드 별로 작업의 결과를 기록해 놓은 로그 파일을 하나의 파일로 병합하는 모듈이다.

3.1.6 디렉토리 처리 모듈

파일들에 대한 처리가 완료된 후 비어 있는 디렉토리를 처리하는 모듈이다. 디렉토리 내에 존재하는 링크파일은 무시하고 삭제를 수행한다.

3.2 구성 파일

파일명	용도
conf.c	환경설정관련 처리를 하는 모듈
idlist.c	아이디와 그룹명을 처리하는 모듈
joblist.c	사용자 작업 관련처리를 하는 모듈
logger.c	로그 파일 기록을 위한 모듈
main.c	쓰레드를 이용하는 메인 프로그램
userlist.c	예외 사용자 처리를 위한 모듈
util.c	기타 공통 처리를 위한 모듈
env.h	공용 변수를 위한 모듈

(표 2) 구성 파일 목록 및 용도

3.3 성능 분석

성능 비교는 실제 파일의 처리(삭제)는 수행하지 않고 삭제할 파일의 목록을 기록하는 것으로 대신하였다. 비교를 위해서 기존에 사용하던 프로그램이 사용한 유닉스의 find 명령어를 이용하여 소요시간을 측정하였으며, 재귀 호출을 이용한 방법과 쓰레드를 이용한 방법을 각각 비교하였다.

운영체제	AIX 5.1L
파일시스템	GPFS

(표 3) 시험 환경

구분	소요 시간(초)	대상 파일수(개)
find 명령어	486.26	313,762
재귀 호출 사용	387.84	
쓰레드(4) 이용	310.67	

(표 4) 성능 비교

성능 시험의 결과를 보면, find 명령어를 통하여 파일을 검색했을 때 보다 재귀 호출 방식으로 구현하였을 경우에는 약 20%정도 성능이 향상되었으며, 쓰레드를 이용하였을 경우 36%의 성능이 향상되었다.

4. 결론

본 고에서는 대용량 파일 관리를 위한 프로그램을 쓰레드를 이용하여 구현하였다. 당초 예상과 달리 쓰레드를 이용하여 파일을 삭제 하였을 경우의 성능 향상이 크게 증가하지는 않았다. 이것은 대부분의 처리가 CPU와 메모리만을 이용하여 처리되는 것이 아니라 디스크에 접근하는 I/O가 주요 처리 내용이기 때문에 디스크 I/O로 인한 병목 현상에 기인한다. 그러나 어느 정도의 성능 향상은 가져온 것은 확인하였으며, 병렬 파일 시스템에서 병렬로 파일을 삭제하는 부분까지 적용한다면 보다 큰 성능의 향상을 가져올 것으로 예상된다.

향후에는 쓰레드를 이용한 병렬 처리 부분을 최적화하여 최적의 성능을 낼 수 있도록 보완을 해야 할 것이다.

참고문헌

[1] 주민규, 권상호, 윤종수 공역 "Unix System Programming", 인포북

[2] 권상호, 고성규 공역, "POSIX(포직스) 쓰레드를 이용한 프로그래밍", 인포북

[3] 정재은, "유닉스 시스템 프로그래밍", 한빛미디어