제 19 회 한국정보처리학회 추계학술발표대회 논문집 제 9 권 제 2 호 (2002.11)

분산 저장시스템에서 가상 트리를 이용한 효율적인 복제 프로토콜

최성춘, 윤희용
성균관대학교 정보통신 공학부
{choisc, youn}@ece.skku.ac.kr

An Effective Replication Protocol using Virtual Tree in Distributed Storage System

Sungchune Choi and Hee Yong Youn
School of Information and Communications, Sungkyunkwan University

요 약

최근 분산 컴퓨팅 환경에서 데이터와 서비스의 복제는 통신비용의 감소, 데이터 가용성 증가, 그리고 단일 서버의 병목현상을 피하기 위해 필수적이다. 기존의 대표적인 복제 프로토콜로 네트워크를 논리적으로 구성하는 Tree quorum 프로토콜이 있다. Tree quorum 프로토콜은 최선의 경우 가장 우수한 읽기 성능을 보이는 반면 트리의 높이가 증가할수록 노드의 수가 기하급수적으로 증가한다는 단점을 가지고 있다. 따라서 본 논문에서는 Tree quorum 프로토콜의 장점을 가지며, 급격한 노드 증가에 따른 성능 저하 문제를 해결하기 위한 가상 트리 프로토콜을 제안한다. 제안된 가상 트리 프로토콜은 안정된 삼각형 구조의 노드 구성과 가상 구조를 통한 Quorum 프로토콜의 사용으로 Tree quorum 프로토콜에 비해 적은 읽기 비용을 가지며, 적은 수의 노드 구성에서도 높은 읽기 가용성을 갖는다.

1. 서론

오늘날 대용량 분산 컴퓨팅 환경에서 데이터와 서비스의 복제는 데이터의 가용성을 높이고, 전체 시스템의 성능을 향상시키기 위해 필수적인 기술이다 [1]. 또한 다중 노드를 사용함으로써 단일 노드의 사용으로 인한 병목현상 문제를 해결할 수 있다. 그러나 노드의 수가 증가할수록 통신비용이 증가하게 되므로, 전체 시스템을 구성하는 노드 중에 읽기/쓰기 동작을 수행하는 노드의 수는 가능하면 적은 수로 유지되어야 한다 [2][3][4].

데이터를 복제하는데 있어 중요한 문제는 여러 노드들에 각기 저장된 데이터의 일관성을 유지하는 것이다. 일관성 제어 프로토콜은 일관된 데이터를 유지하기 위하여 사용자 동작의 동기화를 수행한다. 예를 들어, 각기 다른 사용자로부터 수행되는 쓰기 동작은 복제된 데이터의 동시 변경을 허용하지 않도록 해야 한다 [5]. 이러한 일관성 제어를 위해 기존의 대표적인 방식은 읽기/쓰기 동작을 수행하는데 필요한 노드의 집합을 정의하는 Quorum 프로토콜이 존재한다. Quorum 프로토콜은 읽기 동작을 위한 RQ(read quorum)과 쓰기 동작을 위한 WQ(write quorum)으로 정의된다. 여기서 WQ 과 RQ 은 적어도 하나 이상의 노드를 중복하여 포함해야 항상 최신 데이터에 대한 읽기 동작을 보장할 수 있다.

기존의 대표적인 복제 프로토콜은 Tree quorum 프로토콜이다. Tree quorum 프로토콜 [6]은 노드들의 논리적인 구성을 통하여 전체 노드 중 일부의 노드만을 이용하여 읽기/쓰기 동작을 수행하는 복제 프로토콜이다. 물론 일부 노드만을 사용함으로써 발생하는 일관성 문제를 해결하기 위해 이전에 설명한 Quorum 프로토콜을 사용한다. Tree quorum 프로토콜은 트리의 루트 노드를 이용하여 루트 노드가 장애가 발생하지 않을 경우 1 의 적은 읽기 비용을 가지는 반면 트리의 높이가 증가할수록 노드의 수가 기하급수적으로 증가한다는 단점을 가지므로 더 많은 노드 접근으로 인한 높은 읽기 비용을 가지는 문제점을 가진다.

따라서 본 논문에서는 기존의 Tree quorum 프로토콜이 가지는 장점을 모두 가지면서 Tree quorum 프로토콜의 단점을 해결할 수 있는 가상 트리 복제 프로토콜을 제안한다. 제안된 복제 프로토콜은 안정된 삼각형 구조의 노드 구성과 가상적인 Tree quorum 프로토콜을 이용한 방식으로, Tree quorum 프로토콜에 비해 안정된 읽기 성능을 가지며, 또한 적은 수의 노드 구성에서도 높은 읽기 가용성을 가지게 된다

본 논문의 구성은 다음과 같다. 2 장에서는 기존의 대표적인 Tree quorum 프로토콜에 대하여 설명한다. 3 장에서는 본 논문에서 제안하는 가상 트리 복제 프로토콜을 소개하고, Tree quorum 프로토콜과의 읽기/쓰기 비용 및 가용성을 비교한다. 마지막으로 4 장에서는 논문의 결론을 제시한다.

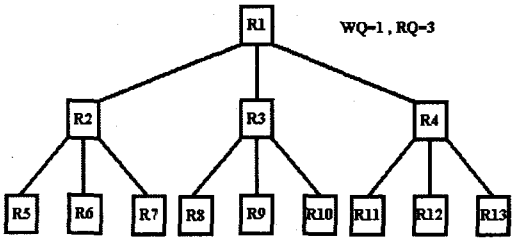
2. Tree quorum 프로토콜

Tree quorum 프로토콜은 (그림 1)과 같은 논리적 트리 구조로 구성된다. 높이 h의 트리로 구성된 n개의 노드가 있다고 가정하면, 잎 노드를 제외한 각 노드 R_i는 S_{R_i}수 만큼의 자식 노드를 갖는다. 각 노드들을 위한 읽기와 쓰기 집합을 정의하기 위해 read quorum rq_{R_i}와 write qrorum wq_{R_i}를 정의한다. Tree quorum 프로토콜은 다양한 rq_{R_i}와 wq_{R_i} 값을 가질수 있으며, 그에 따라 다른 성능을 보인다. 이때 rq_{R_i}의 값이 S_{R_i}의 값과 동일하고, wq_{R_i} 값이 1인 프로토콜을 Logarithmic 프로토콜이라 하며, Tree quorum 프로토콜 에서 가장 좋은 성능을 나타낸다.

알고리즘

읽기 동작 : 루트 노드로부터 시작되며, 만약 루트 노드가 장애가 발생하면 루트의 rq_{R_i}를 읽게 된다. rq_{R_i}에 포함된 자식 노드는 다시 루트 노드와 같이 동작하게 된다.

쓰기 동작 : 루트 노드와 루트의 wq_{R_i}를 쓰게 된다. 선택된 wq_{R_i}는 다시 루트 노드와 같이 동작하게 된다.



(그림 1) 13개의 노드를 갖는 높이 3의 트리

Tree quorum 프로토콜중 Logarithmic 프로토콜의 읽기와 쓰기 예는 다음과 같다. 읽기 동작을 위해 필요한 노드들의 집합 RQ는 {R1}, {R2, R3, R4}, {R3, R4, R5, R6, R7}, 그리고 {R3, R5, R6, R7, R11, R12, R13}이 된다. 쓰기 동작을 위해 필요한 노드들의 집합 WQ는 {R1, R2, R6}, {R1, R3, R8}, 그리고 {R1, R4, R11}이 된다.

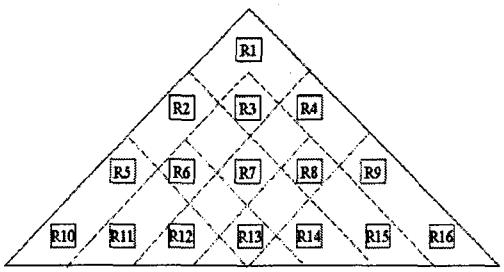
3. 제안된 프로토콜

이전 장에서 설명한 것과 같이 Tree quorum 프로토콜은 몇가지 문제점들을 가지고 있다. 따라서 본 논문에서는 루트 노드의 고장이 없을 경우 우수한 읽기 성능을 가지는 Tree quorum 프로토콜의 장점을 가지면서 노드의 급격한 증가 문제를 해결하며, Tree quorum 프로토콜에 비해 낮은 읽기 비용과 높은 읽기 가용성을 가지는 새로운 가상 트리 복제 프로토콜을 제안한다.

3.1 가상 트리 복제 프로토콜

대부분의 복제 프로토콜과 동일하게, 본 논문에서 제안한 프로토콜에서는 노드의 고장은 발생 할 수 있지만 비잔틴 고장은 고려하지 않는다.

제안된 프로토콜은 (그림 2)과 같이 안정된 삼각형의 구조를 이용하여 노드들이 구성되며, 읽기와 쓰기 동작은 기존의 quorum 프로토콜과 동일하게 동작하게 된다. Tree quorum 프로토콜과의 차이점은 Tree quorum 프로토콜은 각 부모 노드가 동일한 개수의 독립된 자식 노드를 가지는 반면, 제안된 가상 트리 프로토콜에서의 상위 노드는 가상의 노드 구성을 통해 중첩된 하위 노드들을 가지게 된다. 이러한 자식 노드의 중첩 사용으로 인하여 레벨이 증가함에 따라 노드의 수가 기하급수적으로 증가하는 문제점을 해결할 수 있으며, 노드의 실패에 따른 읽기 비용 역시 감소하게 된다. 제안된 프로토콜의 노드 구성은 (그림 2)에서와 같이 각 노드의 자식 노드는 점선에 의하여 구분되며, 실제로 (그림 3)에서 보는것과 같이 인접한 노드는 2 개의 자식 노드를 공유하여 사용하게 된다. 예를 들어, R2의 자식노드는 R5, R6, 그리고 R7 이 되며, R3 의 자식 노드는 R6, R7, 그리고 R8 이되어, R2 와 R3 는 R6 와 R7 의 자식 노드를 중복하여 가지게 된다.



(그림 2) 높이 4 의 제안된 가상 트리 구조

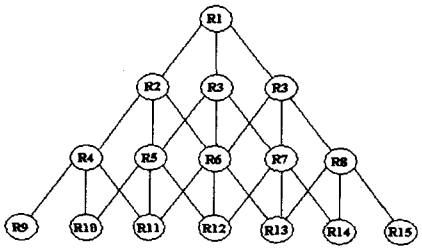
□ *읽기 동작*

읽기 동작은 루트 노드로부터 시작하고, 만약 루트노드가 실패할 경우에는 자식 노드 전체에 대하여 읽기를 수행한다. 루트의 자식 노드는 다시 루트와 같이 동작하게 되며, 트리의 끝에 도달할때까지 반복하여 동작한다.

□ *쓰기 동작*

쓰기 동작은 읽기 동작과 마찬가지로 루트 노드로부터 시작하고, 루트의 자식 노드중에 임의의 하나의 노드를 선택하여 쓰기 동작을 수행한다. 선택된 루트의 자식 노드는 루트와 같이 동작하게 되고, 트리의 끝까지 반복하여 동작하게 된다.

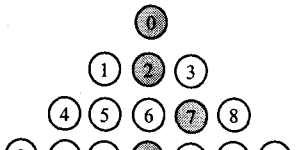
(그림 3)에서 읽기 동작을 위한 가능한 노드들의 집합은 {R1}, {R2, R3, R4}, {R3, R4, R5, R6, R7}, 그리고 {R4, R5, R6, R7, R8}이 된다. 쓰기 동작을 위한 가능한 노드들의 집합은 {R1, R2, R6, R11}, {R1, R3, R6, R12}, 그리고 {R1, R4, R9, R16}이 된다.



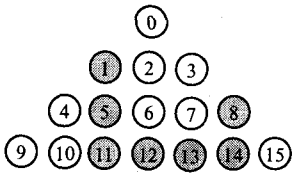
(그림 3) 가상 트리 구조의 논리적 연결

3.3 제안된 프로토콜의 동작 예

(그림 4)는 제안된 가상 트리 복제 프로토콜의 동작 예를 보여주고 있다. (그림 4)의 예는 16개의 노드를 갖는 레벨 4의 구조에서 읽기 동작과 쓰기 동작을 나타낸다. (그림 4-a)는 3.2절에서 설명된 알고리즘에 따라 쓰기 동작을 수행한다. (그림 4-b)의 읽기 동작에서는 루트 노드가 실패하고, 차례로 2, 3, 6, 그리고 7번노드가 실패한 경우의 읽기 동작에 참여한 노드를 보여주고 있다.



(a) 쓰기 동작 : WQ = {R0, R2, R7, R12}



(b) 읽기 동작 : RQ = {R1, R5, R8, R11, R12, R13, R14}

(그림 3) 제안된 프로토콜의 읽기 및 쓰기 동작

제안된 가상 트리 복제 프로토콜 동작의 정확성은 다음의 정리와 증명를 통하여 보장된다.

정리 1: 읽기 동작은 항상 마지막 쓰기 동작에 의해 저장된 최신의 값을 읽게된다.

증명. (그림 3)에서 보는것과 같이, 임의로 선택된 RQ 과 WQ 을 위해, $WQ \cap RQ \neq 0$ 를 만족한다. 그러므로 임의로 선택된 RQ 은 항상 최신 데이터를 포함한다는 것을 보증할 수 있다.

정리 2: A 읽기/쓰기 또는 쓰기/쓰기 동작은 동시에 수행될 수 없다.

증명. · · · · · · · · · RQ · WQ · · · · · ·
· ·
· ·
· ·

3.3 성능평가

이번 절에서는 제안된 프로토콜의 읽기 및 쓰기 비용과 가용성에 대하여 기존의 Logarithmic 프로토콜과 비교하여 한다. 정확한 성능평가를 위하여 다음과 같은 전제를 사용한다.

- 링크의 실패는 발생하지 않는다.
- 노드들의 실패는 독립적이며 실패율은 일정하다.
- 트리 구조에서 잎 노드를 제외한 각 노드들은 같은 수의 자식을 갖는다.

3.3.1 비용 분석

비용은 전체 노드들중에 읽기와 쓰기 동작을 수행하는 노드의 수를 의미한다. 제안된 프로토콜에서 모든 읽기/쓰기 동작이 루트로부터 시작되기 때문에 루트 노드의 병목현상이 발생할 수 있다. 이러한 문제를 해결하기 위해 읽기 동작을 임의의 레벨에서 시작하도록 변경해준다. 이를 위해 읽기 동작을 수행하기 전에 다음의 동작이 추가 된다. 균일 밀도 함수에 의한 [0,1]의 값을 갖는 랜덤 변수 f를 선택한다. [0,1]의 값을 갖는 랜덤 변수 x를 발생시키고, 만약 $x \le f$ 조건을 만족하면 루트노드에 대한 읽기 동작을 수행한다. 그렇지 않을 경우, 다시 랜덤 변수 x를 발생시켜 다음레벨에 대하여 동일한 동작을 수행하게 되어, 읽기 동작을 수행할 레벨이 결정되게 된다.

제안된 프로토콜의 읽기 비용은 다음과 같이 계산되며, 여기서 f는 각 레벨이 선택될 확률이다.

$$C_{read} = f + (1-f)\cdot f\cdot 3 + (1-f)^2\cdot f\cdot 5 + \cdots + (1-f)^{h-1}\cdot f\cdot (2(h-1)+1) + (1-(f + (1-f)\cdot f + (1-f)^2 + \cdots + (1-f)^{h-1}\cdot f))\cdot (2h-1)$$

$$= f\sum_{k=0}^{h-1}(1-f)^k\cdot(2k+1) + \left(1 - f\sum_{k=0}^{h-1}(1-f)^k\right)\cdot(2h-1)$$

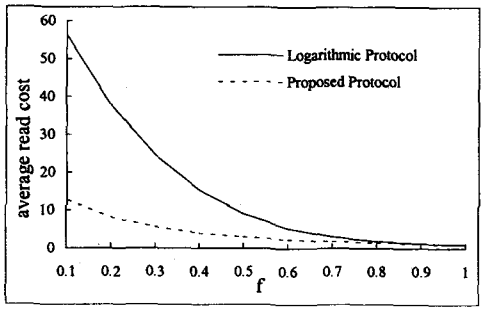
최소 읽기 비용은 루트노드에 대한 읽기 동작이 성공한 경우이므로 다음과 같다.

$$C_{read(\min)} = 1$$

제안된 프로토콜을 위한 쓰기 비용은 각 레벨에서 하나의 노드에만 쓰기를 수행하기 때문에 레벨에 의존하게 된다. (그림 4)는 121개의 노드를 갖는 Logarithmic프로토콜과의 읽기 비용을 비교한 것이다.

(그림 4)에서 보는것과 같이 같은 수의 노드를 가질 때 평균 읽기 비용은 제안된 프로토콜이 Logarithmic프로토콜에 비해 훨씬 우수한 것으로 나타난다. 이러한 결과를 보이는 이유는, 121개의 노드를 가지기 위해서 Logarithmic 프로토콜의 경우는 잎노드의 수가 81개로 구성되지만, 제안된 프로토콜의 경우는 마지막 레벨이 21개의 노드만으로 구성 되기 때문에, 최악의

경우 읽기 비용이 Logarithmic 프로토콜 보다 적게 된다.



(그림 4) 121개의 노드에서 읽기 비용 비교

3.3.2 가용성 분석

복제 프로토콜에서의 가용성은 노드의 실패에도 불구하고 성공적인 동작을 의미한다. 따라서 동일한 노드 실패율에서 높은 가용성을 보이는 복제 프로토콜이 성능면에서 더욱 우수하다.

일반적으로 여러 저장 노드의 가용성은 다음 식과 같이 2 항(*binomial*) 함수에 의하여 정의 할 수 있다. 여기서 S 와 p 는 자식 노드의 수와 노드들의 성공 확률을 각각 의미한다. 즉 2 항 함수는 성공적으로 동작을 수행하기 위해 필요한 노드의 전체 조합을 의미한다.

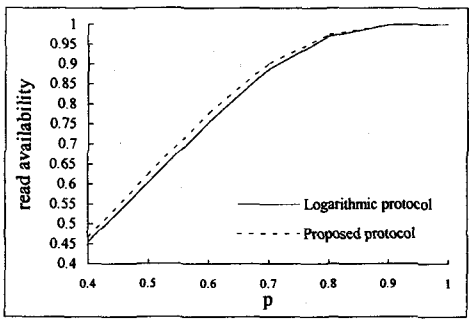
$$avail = \sum_{k=0}^{S} \binom{S}{k} p^k (1-p)^{S-k}$$

Logarithmic 프로토콜은 각 부모노드가 독립된 자식 노드를 가지를 반면 제안된 가상트리 프로토콜은 자식 노드를 공유하므로 부모 노드의 실패 조건에 따라 읽기 동작을 수행해야 하는 자식 노드의 수가 틀려지게 된다. 예를 들어 (그림 3)의 레벨 2 에서, R2 와 R3 노드의 장애가 발생할 경우, 즉 인접한 부모 노드에 장애가 발생할 경우는 4 개의 자식노드(R5, R6, R7, R8)에 대하여 읽기 동작을 수행한다. 하지만 만약 R2 와 R4 노드에 장애가 발생할 경우, 5 개의 자식 노드 전체에 대하여 읽기 동작을 수행하게 된다. 이와 같이 부모 노드의 실패 조건에 따라 읽기 동작에 참여하는 자식 노드의 수가 결정되게 되며, 이것은 하위 레벨로 갈수록 더욱 많은 경우를 가지게 된다. 간단하게 레벨 3 으로 구성된 가상 트리의 가용성을 분석하면 다음식과 같다.

$$\wp_{level3} = p + (1-p)\left(p^3 + 3p^2(1-p)p^3 + p(1-p)^2(2p^4 + p^5) + (1-p)^3 p^5\right)$$

(그림 5)는 같은 레벨에서 제안된 프로토콜과 Logarithmic 프로토콜의 읽기 가용성을 비교한 결과이다. 두 프로토콜이 같은 레벨로 구성될 때, Logarithmic 프로토콜은 40 개의 노드를 가지는 반면 제안된 가상 트리 프로토콜은 16 개의 노드만을 가지게 된다. 적은 수의 노드로 구성됨에도 불구하고 제안된 프로콜의 읽기 가용성은 Logarithmic 프로토콜에 비해 좋게 나타난다. 이것은 가상 트리 프로토콜이 더 적은 노드 수의 RQ 을 가지기 때문이다.

제안된 가상 트리 프로토콜의 쓰기 가용성은 Logarithmic프로토콜과 같은 결과를 가져온다.



(그림 5) 읽기 가용성 비교

4. 결론 및 향후 계획

본 논문은 Logarithmic프로토콜과 논리적으로 같은 구성을 가지면서, 물리적으로 더 적은 노드의 구성이 가능한 가상 트리 복제 프로토콜을 제안한다. 제안된 프로토콜은 Logarithmic프로토콜에 비해 우수한 읽기 비용 및 가용성을 보인다. 일반적인 저장 장치 시스템이 80%의 읽기 동작과 20%의 쓰기 동작을 수행하므로, 제안된 프로토콜은 기존의 *Logarithmic*프로토콜에 비해 더욱 우수한 성능을 가져올 수 있게 된다.

향후 제안된 프로토콜의 보다 정확한 성능 평가를 수행하기 위해 시뮬레이션을 수행하고, 이를 통해 응답시간과 처리율을 비교할 계획이다.

참고문헌

[1] C. Amza,, A. L. Cox, W. Zwaenepoel, Data replication strategies for fault tolerance and availability on commodity clusters, *Proceedings International Conference on Dependable Systems and Networks (DSN)*, 2000, 459-467.

[2] H.Y. Youn, D. Lee, B. K. Lee, J. S. Choi, and H. G. Kim, An Efficient Hybrid Replication Protocol for Highly Available Distributed System, *Proceedings IASTED on Communications and Computer Networks (CCN)*, Nov, 2002.

[3] D. Saha, S. Rangarajan, S. K. Tripathi, An Analysis of the Average Message Overhead in Replica Control Protocols, *Proceedings IEEE Transactions on Parallel and Distributed Systems*, 7(10), Oct, 1996, 1026-1034.

[4] G. Alonso, Partial Database Replication and Group Communication Primitives, *Proc. of the 2nd European Research Seminar on Advances in Distributed Systems (ERSADS'97)*, March 1997, 171-176.

[5] T. Anderson, Y. Breitbart, H. Korth, A. Wool, Replication, Consistency, and Practicality: Are These Mutually Exclusive?, *Proceedings ACM SIGMOD International Conference on Management of Data*, Jun 1998, 484-495.

[6] D. Agrawal and A. El Abbadi, The tree Quorum protocol: An Efficient Approach for Managing Replicated Data, *Proceeding 16th Very Large Databases (VLDB) Conference*, 1990, 243-254.