논문 2021-2-3 http://dx.doi.org/10.29056/jsav.2021.12.03

소규모 노드로 구성된 고속 병렬 블록체인 아키텍처

조용준*† , 신동명*

Concurrent blockchain architecture with small node network

YongJoon Joi*† , DongMyung Shin*

요 약

블록체인 기술은 장점인 신뢰성 문제를 넘어, 산업에서 요구하는 각종 성능을 만족하기 위한 단계에 접어들었다. 하지만, 블록체인 아키텍처의 특성이 걸림돌이 되어, 반응성 및 병렬 확장성 개선에 어려움을 겪고 있다. 블록체인 기술을 산업에 적용하기 위해서는 성능 문제를 해결할 수 있도록 아키텍처를 재설계해야 한다.

본 연구에서는 블록체인의 기술적 특징을 보존하면서, 동시에 병렬처리 성능 및 반응성 향상을 위한 새로운 요소 기술과 이를 통합한 아키텍처 TPAC를 개발함으로써, 안정적이면서 빠른 트랜잭션 처리, 저지연성 등, 다양한 면에서의 성능이 개선됨을 보였다.

Abstract

Blockchain technology fulfills the reliance requirement and is now entering a new stage of performance. However, the current blockchain technology has significant disadvantages in scalability and latency because of its architecture. Therefore, to adopt blockchain technology to real industry, we must overcome the performance issue by redesigning blockchain architecture.

This paper introduces several element technologies and a novel blockchain architecture TPAC, that preserves blockchain's technical advantage but shows more stable and faster transaction processing performance and low latency.

한글키워드 : 블록체인, 병렬처리, 분산 컴퓨팅, 트랜잭션, 소프트웨어 트랜잭셔널 메모리

keywords : blockchain, concurrent processing, distributed computation, transaction, STM(software transactional memory)

1. 서 론

블록체인 기술은 검증 단계를 넘어 신뢰성과 투명성을 요구하는 다양한 산업에서 그 적용 범위를 넓혀가고 있다.

하지만, 산업에서는 신뢰성과 투명성만이 아니라, 성능 또한 요구한다. 특히 최대 성능보다 안정적인 운용을 위해 어떠한 상황에서도 달성 가능한 최저 보장 성능이 더 중요한 때도 있다. 하지만, 블록체인이 구조적으로 담보하는 신뢰성·투명성에 비해 안정적인 성능 보장은 블록체인의

* 엘에스웨어(주)
† 교신저자: 조용준(email: research@holos.works)
접수일자: 2021.12.02. 심사완료: 2021.12.13.
게재확정: 2021.12.20.

www.kci.go.kr

난제 중 하나로, 산업에 적용하지 못하는 대표적인 걸림돌이 되고 있다.

블록체인 기술의 산업 적용을 위한 목표를 정리해보면 다음과 같다. 1. 의존성 있는 트랜잭션 처리에 대한 안정적인 성능 보장, 2. 노드 증설에 따른 성능 확장성 보장, 3. 원장 데이터에 대한 빠른 접근 속도 제공 등이다.

분산 구조에서의 일관성 문제는 병렬·분산 컴퓨팅의 가장 중요한 문제 중 하나이다. 초창기 블록체인에서는 이 문제를 해결할 필요가 없었는데, 하나의 리더 노드가 생성한 블록을 나머지 노드가 검증만 하므로, 일관성 문제가 발생하지 않는다.

블록체인의 분산 노드 구조는 안정적인 성능 보장도 어렵게 한다. 블록체인은 여러 개의 분산 노드로 구성되나, 각 노드는 분산처리가 아니라 노드 간 상호 검증을 위해 활용되기 때문에, 노드 숫자가 병렬처리에 직접적으로 도움이 되지는 않는다. 병렬처리가 가능한 예외를 제외하고, 블록체인의 최대 성능은 노드 규모와 관계없이 노드 한 개의 처리 능력으로 제한되어왔다. 노드 증설에 의한 블록체인 성능 확장성 문제, 즉 병렬처리 성능 향상을 위한 방법으로 Ethereum 2.x의 layer-2 구조[11] 등이 있으나 2.3에서 자세히 설명하는 한계점을 지니고 있다. Caper, Fabric++, XOX Fabric, NexLedger Accelerator와 같은 연구는 노드 단위 병렬처리는 가능하지만, 비확정적인 키를 다루지 못하거나 비효율적인 구조를 도입하는 등, 기술적으로 충분히 성숙하지 못하였다.

또한, 블록체인은 랜덤 액세스 성능이 낮은 순차 기록구조를 채택하고 있다[10]. 이 구조는 블록체인의 신뢰성을 담보하는 핵심 요소로, 데이터베이스와는 달리 어떤 키의 최종값만을 기록하는 것이 아니라 모든 중간 상태도 보관한다. 하지만, 어떤 키의 최신 값이 어떤 블록에 기록되어있는지 알 수 없으므로, 블록체인 서비스 운영을 지속하면 원장 기록이 계속 커질 뿐만이 아니라, 값 탐색 시간도 길어진다.

본 논문에서는 상호 의존성 분석을 통해, 트랜잭션을 병렬처리 가능한 집합으로 분류하고, 노드 그룹 단위로 트랜잭션 집합을 병렬로 처리, 그 결과를 물리적으로 병렬 기록할 수 있는 요소기술 개발을 통해 블록체인의 병렬처리 효율 향상을 도모하고, 이를 통합한 블록체인 아키텍처 TPAC(Transaction Proposal Aggregation for Concurrency)을 개발하였다. 5장에서 기존 블록체인 아키텍처와 TPAC의 성능 차이를 검증한다.

2. 기존 블록체인 아키텍처와 성능 문제

2.1 병렬처리와 키 충돌 문제

병렬처리를 통해 성능을 향상하려는 경우, 종속성과 일관성(키 충돌 문제)으로 구분해 볼 수 있다. 블록체인에서는, 각 트랜잭션 내의 처리를 병렬화하는 것은 아키텍처 상에서 고려할 문제가 아니다. 하지만, 일관성은 블록체인의 성능에 직접적인 영향을 끼치기 때문에, 중요한 문제이다.

블록체인과 기존 병렬·분산 컴퓨팅이 일관성 유지 측면에서 어떻게 차이가 있는지, 상태 갱신(state update) 주기와 갱신 방식, 그리고 노드 구조의 측면에서 살펴본다.

2.2 일관성 문제

일관성(키 충돌) 문제를 해결하기 위하여, 1970년대부터 다양한 병렬처리 기법이 제안되어 왔다. 대표적인 기법은 semaphore 등의 lock-based strategy이다. 하지만, lock-based strategy는 lock을 관리하는 컴포넌트와의 통신 지연 및 신뢰성 등으로 인해 분산 노드 네트워크인 블록체인 아키텍처에 적합한 구조가 아니다.

www.kci.go.kr

lock-free strategy는 1980년대부터 등장한 아이디어로, STM(Software Transactional Memory)[12] 또는 RLU(Read-Log-Update)가 가장 대표적인 기법이다. Lock-free strategy는 낙천적 접근 방식(병렬처리 간에 간섭이 발생할 가능성이 낮은 경우에 효율적)으로, 원자적(atomic)으로 처리되어야 하는 계산(트랜잭션)을, 계산 과정에서 읽은 키와 쓰는 키의 목록을 기억하고 실행이 완료되면 관리되는 키가 실행 기간 중에 다른 트랜잭션과 간섭(충돌)하지 않았는지를 확인한다. 간섭이 없는 경우에는 계산 결과에 의한 변경 값을 반영하고, 간섭이 있는 경우에는 변경 값을 반영하지 않는(트랜잭션 취소) 방식을 취한다. 데이터베이스 및 블록체인을 포함하는 트랜잭션 시스템이 이와 같은 방식을 채택하고 있다.

2.3 트랜잭션 실행 구조와 병렬처리

블록체인 아키텍처 중 Hyperledger Fabric[8]은 트랜잭션 단위로 STM 방식과 유사한 방식으로 블록을 생성하며, Ethereum이나 Bitcoin 등 대부분의 블록체인은 트랜잭션이 아닌 블록 단위로 STM 방식을 적용하고 있다.

하지만, 블록체인은 일반적인 병렬·분산 컴퓨팅에 비해 상태 갱신 주기가 길어서, STM 방식은 비효율적일 수 있다. 충돌이 발생하는 경우 바로 그 트랜잭션을 재실행 또는 취소한다. 이를테면, 실행하는데 10ms가 걸리는 트랜잭션의 경우, 병렬처리된 모든 트랜잭션이 충돌하더라도 약 1초마다 100개 정도를 처리할 수 있다. 그런데, 블록체인에서는 상태 갱신 주기를 기준으로 충돌을 확인한다. 즉, 상태 갱신 주기(블록 생성 주기)가 1초인 블록체인에서 모든 트랜잭션이 충돌하는 경우, 1초(한 블록)당 최대 1개씩만 처리 가능, 즉 1TPS보다 낮은 성능밖에 달성할 수 없는 한계를 지니고 있다. 이러한 구조로 인하여, 블록체인은 충돌이 발생할 수 있는 요청을 병렬로 처리하면 실제 수행한 트랜잭션 계산량에 비해 유효 트랜잭션 비율이 너무 낮아서, 실질 처리 성능이 낮아질 수 있다[6].

이로 인하여, 트랜잭션 단위 병렬처리가 가능한 Hyperledger Fabric의 경우, 이상적인 경우의 최대 성능과 무관계하게 보장 가능한 최소 성능(보장 TPS)은 매우 낮다[6]. 기존의 Ethereum과 Bitcoin 등에서는 트랜잭션 실행(블록 생성)을 단일 노드에서 수행하기 때문에 처음부터 병렬처리를 하지 않는다[1]. Ethereum 2.x부터 도입되는 Layer-2 구조는 layer간 의존성이 없는 트랜잭션을 각 layer에서 단독으로 처리함으로써 처리 성능 향상을 꾀하고 있다. Ethereum 전체로 보면 layer가 증가할수록 성능이 향상되는 것처럼 보이나, 이는 의존성이 연산을 독립적인 시스템(layer-2)에서 각자 처리한 성능을 합산한 것을 Ethereum의 성능으로 표현한 것으로, 실질적으로 독립적인 블록체인을 구성하는 개별 layer의 성능이 직접적으로 향상되는 것은 아니다.[2]

결과적으로, 블록체인의 각 분산 노드에서 병렬로 트랜잭션을 처리하더라도, 독립적으로 수행된 트랜잭션이 충돌하는 경우 실제 성능은 최대 성능치와 무관한 낮은 성능밖에 달성하지 못하는 문제가 발생한다.

---

1) Ethereum도 한 노드 안에서 트랜잭션을 병렬처리할 수 있으나, 트랜잭션 검증 때 각 트랜잭션을 독립적(병렬)으로 검증할 수 있도록 의존성이 있는 트랜잭션은 하나의 블록 안에 담지 않는다.

2) layer 간 의존성이 있는 트랜잭션은 하나의 layer 안에서 독립적으로 실행할 수 없다. 이러한 트랜잭션을 포함하는 블록은 layer-1 에서 충돌할 수 있다. 즉, 충돌한 트랜잭션을 포함한 layer-2 블록체인 안의 전체 트랜잭션을 파기하고, 재실행 또는 실행 취소를 해야만 한다.

www.kci.go.kr

3. 블록체인을 위한 병렬실행 기술

본 연구에서는 위의 대규모 병렬 시뮬레이션을 위한 기술을 블록체인에 응용하여, TPAC 아키텍처를 개발하였다.

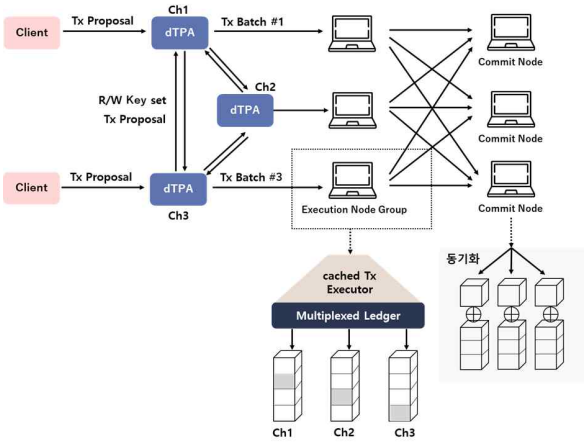

그림 1. TPAC 아키텍처 구조도
Fig. 1. TPAC architecture diagram

TPAC 아키텍처는 크게 RWKS 정보 기반 트랜잭션 분류 기술인 TPA, 의존관계가 있는 트랜잭션 실행을 위한 트랜잭션 실행기, 그리고 병렬 원장과 One-Pass Prefetching 기술으로 구성된다. 3장에서는 TPA, 4장에서 One-Pass Prefetching 기술을 설명한다.

3.1 RWVS 정보 기반 병렬 시뮬레이션 기술

Bottom-up 방식 시뮬레이션의 경우, 시뮬레이션 환경 안에서 수많은 구성요소가 독립적으로 판단하고 행동한다. 이러한 시뮬레이션을 대규모로 확장하는 경우, 병렬처리 기술은 필수적이다. 그런데, 각각의 구성요소의 판단 또는 행동 결과가 상호 간섭으로 인해 변할 가능성이 있는 경우, 각 구성요소의 시뮬레이션을 무조건적으로 각각 실행하는 것은 다양한 에러를 발생시킬 수 있다. 이러한 문제를 해결하기 위해서는, 어떤 구성요소가 간섭 가능성이 있는지를 분석하고 간섭 여부에 따라서 독립적으로 실행 가능한지, 아니면 순차적으로 실행할 것인지를 고려해야 한다

Holos Works에 개발한 대규모 병렬 시뮬레이터인 CoMPleT Engine은 각 구성요소의 판단 및 행동 원칙을 개별 스크립트로 작성하고, 동일 시간대에 실행할 스크립트가 생성·조회·수정하는 변수를 분석하여 스크립트 간의 의존성을 분석하는 기술(Spool Synthesizer)을 개발하였다. 이 기술은 스크립트를 런타임 중에 분석하여 RWVS (Read/Write Variable Set)이라 불리는, 생성·조회·수정할 가능성이 있는 변수의 집합을 생성하여 병렬처리의 일관성 문제를 해결하였다.

어떤 두 개의 스크립트가 의존성이 있는 (RWVS가 교집합을 가지고 있는) 경우, 이 두 스크립트를 병렬(독립적)로 실행하면 일관성이 깨질 수 있다. Spool Synthesizer는 RWVS 정보를 기준으로 스크립트를 분류하여 같이 실행해야 할 스크립트와 독립적으로 실행할 수 있는 스크립트를 분류, 분류된 합성 스크립트를 병렬실행하는 인터프리팅 및 가상머신 기술을 개발하였다.

위의 시뮬레이션 기술의 1. 상태 갱신 주기를 블록체인의 블록 생성 주기에, 2. 스크립트를 스마트 컨트랙트에 대응시켜 블록체인 기술에 적용하였다. 즉, Spool Synthesizer에 기반한 TPA, 가상머신 및 인터프리터에 기반한 트랜잭션 실행기를 만들어 TPAC 아키텍처를 개발했다.

3.2 RWKS 정보 기반 트랜잭션 분류 기술

블록체인도 각 스마트 컨트랙트에 의해 실행되는 트랜잭션이 어떤 키의 값을 읽고 쓸 것인지를 예측할 수 있다. 특히 분산 노드 구조로 인해 무작위 값을 사용하기 어려운 블록체인의 특성상, 트랜잭션 요청(transaction proposal)의 입력값과 실행 당시의 원장 상태(global state)가 결

www.kci.go.kr

정되면 읽고 쓰는 키, 즉 트랜잭션 간의 의존성은 예측할 수 있다.

다만, 트랜잭션을 실행해보지 않으면 실제로 사용되지 않는 키, 또는 실행 과정에서 접근하는 키를 로직을 통해 생성하는 경우, 또는 배열과 같이 다수의 변수를 참조하는 등, 사용 가능성은 있지만, 확실히 사용되는 것을 보장할 수 없는 비결정적인 키(uncertain key)를 포함하는 트랜잭션 또한 존재한다.

예측 가능한 RWK(Read/Write Key: 읽고 쓰는 키) 정보를 활용하여, 트랜잭션을 병렬처리하는 선행 연구들로서 Caper[2], Fabric++[6], XOX Fabric[3], NexLedger Accelerator[7], 특허[13] 등이 있다. 이들은 공통으로 1. 트랜잭션 요청으로부터 읽고 쓰는 키(Read/Write Key)를 분석하여 트랜잭션 간의 의존성을 분석하고 2. 상호 의존성이 없는 트랜잭션 배치(batch)를 만들어, 3. 각 노드 또는 노드 그룹에서 실행한 결과를 취합하여 블록을 확정한다.

NexLedger Accelerator는 트랜잭션 간의 의존성을 배척하는 기존 블록체인과 가장 유사한 접근 방식을 채택하였다. 상호 의존성이 없는 트랜잭션 배치를 생성하여 병렬실행, 그 결과를 취합하였다. 이 구조는 NexLedger Accelerator가 기반하고 있는 Hyperledger Fabric과 높은 호환성을 지니고 있으나, 트랜잭션 배치 간의 충돌을 제어하지 않기 때문에, 기존 Hyperledger Fabric 대비 약 2배 정도의 성능 향상만이 가능하다.

XOX Fabric 및 특허[13]는 비결정적인 키를 배제하기 위하여, 일단 트랜잭션을 실행하여(사전 실행) 트랜잭션이 실제로 사용하는 키 목록을 획득, 이 RWKS(Read/Write Key Set) 정보를 활용하여 트랜잭션을 pre-ordering 한 후, 병렬실행하는 구조를 채택하였다.

위와 같은 접근 방식은 비결정적인 키의 상태를 확정한 다음에 분류한다는 점에서 분류에 사용되는 키 정보를 줄일 수 있어서 트랜잭션 분류 단계의 효율을 올릴 수 있다. 하지만, 트랜잭션을 사전 실행하는 동안 블록체인이 정지해있을 수 없으므로, 다른 트랜잭션이 실행되고 그 결괏값이 원장에 반영된다. 즉, 사전 실행 후 분류된 트랜잭션을 실제로 실행하는 단계에서 해당 트랜잭션이 참조하던 값이 변경될 수 있다. 그로 인하여 분류단계에서는 충돌이 발생하지 않으리라고 예측하였던 트랜잭션이 충돌을 일으킴으로 인해, 해당 트랜잭션을 포함하는 트랜잭션 배치 전체의 실행 내용을 파기해야 하는 문제가 발생할 수 있다. 그리고, 사전 실행 구조로부터 쉽게 유추할 수 있듯이, 트랜잭션을 두 번 실행 후, 다시 검증을 수행해야 때문에, 블록 확정까지 총 세 번 순차적으로 실행해야 한다. 즉, 시간 및 에너지 효율 면에서 큰 단점을 지니고 있다.

Caper 및 Fabric++는 결정적인 키만을 고려하는 병렬실행형 블록체인이다. 비결정적인 키를 포함하는 트랜잭션을 실행할 수 없다는 단점을 제외하면, 3.3의 TPA와 유사한 트랜잭션 분류 및 배치 생성 기술을 통한 병렬처리 방식을 도입하고 있다.

3.3 TPA(Transaction Proposal Aggregator)

제안 아키텍처 TPAC의 TPA는 비결정적인 키를 포함하는 트랜잭션과 결정적인 키를 포함하는 트랜잭션을 사전 실행 없이도 분류할 수 있는 [1] CoMPleT Engine의 Spool Synthesizer의 구조에 기반하고 있다.

Spool Synthesizer는 세 가지의 아이디어로 구성된다. 1. 사용될지 확정되지 않은 키는 모두 RWKS에 포함한다. 2. 부분적으로 결정되어있는 비결정적인 키(부분 키)를 결정된 부분을 활용하여 분류하고 3. 완전하게 비결정적인 키를 포함하는 트랜잭션은 모든 트랜잭션 배치의 선두에 포함한다.

www.kci.go.kr

1번과 2번 아이디어는 비효율적으로 보일 수 있으나, 시간적·지리적 국소성에 의한 캐시 효과를 고려하는 경우, 사전 실행을 통해 결정적인 키를 모두 확정하는 것보다 더 높은 효율을 보일 수 있다.

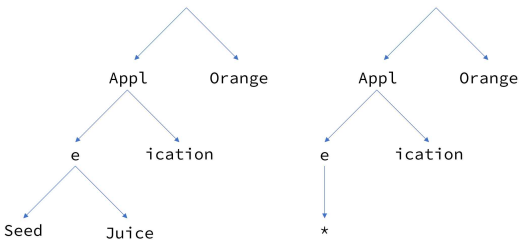

그림 2. RWKS-α, RWKS-β의 예시
Fig. 2. Example of RWKS-α and RWKS-β

비결정적인 키를 다루는 예로서, 그림 2와 같은 두가지 RWKS를 준비한다. 하나는 RWKS-α {“AppeSeed”, “AppleJuice”, “Application”, “Orange”}, 또 하나는 비결정적인 키 “Apple*”를 포함하는 RWKS-β {“Apple*”, “Application”, “Orange”}이다.

결정적인 키의 경우에는 RWKS-α와 같은 집합에서 키의 포함 여부를 확인하면 된다. 이를테면, “AppleJam”은 RWKS-α에 포함되는지 아닌지가 명확하다. 하지만, 비결정적인 키 “App*”의 경우, 집합 방식으로는 포함 여부를 판단하기 곤란하다.

이러한 경우를 다루기 위하여, TPA는 trie[14]와 같은 데이터 구조를 사용하여 비결정적인 키에 대응하고 있다. 이는 배열이나 합성키와 경우에 비슷하게 적용할 수 있어 Caper나 Fabric++가 다룰수 없고, XOX Fabric이나 특허[13]에서 사전 실행을 필요로 하는 유형의 트랜잭션을 사전 실행 없이 다룰 수 있게 된다.

4. 원장 구조와 One-Pass Prefetching 기술

4.1 블록체인의 원장 구조

블록체인은 랜덤 액세스가 어려운 순차 기록 구조를 채택하고 있다. Hash 값으로 연결된 순차 기록구조는 블록체인 원장의 신뢰성을 담보하는 핵심 요소로, 데이터베이스와는 달리 어떤 키의 최종값만을 기록하는 것이 아니라 모든 중간 상태도 보관한다. 하지만, 어떤 키의 최신 값이 어떤 블록에 기록되어있는지 알 수 없으므로, 해당 키의 최신 값을 찾을 때까지 최신 블록으로부터 주사(走査)해야 하므로, 원장이 커지면 반응 속도가 느려질 수 있다. 이는 현존하는 대부분의 블록체인([2][3][6][7])이 공통으로 가지고 있는 문제이다.

이러한 문제를 해결하기 위해 일부 블록체인에서는 원장으로부터 인덱스 정보를 별도 구축하는 방식 등으로 성능 향상을 위한 방법을 제시하고 있다. 하지만, 이는 키의 숫자가 데이터베이스로 관리하기에 적합한 숫자일 때에만 가능하다. 블록체인에 들어있는 모든 키의 값을 데이터베이스에 포함하지 않는 경우, 결국 원장에서 값을 읽어와야 하므로, 원장에 대한 값 읽기 성능을 향상할 방법이 필요하다.

TPAC 아키텍처에서는 이러한 문제를 해결하기 위하여, 병렬 원장 구조 및 One-Pass Prefetching 기술을 도입하였다. 본 논문에서는 다른 블록체인 아키텍처도 응용 가능한 One-Pass Prefetching 기술만 설명한다.

4.2 One-Pass Prefetching 기술

스마트 컨트랙트 실행 시, 여러 개의 읽기 요청을 처리하기 위해 동일 블록을 반복적으로 주사하는 문제를 해결하기 위하여, One-Pass Prefetching 기술[5]은 TPA의 트랜잭션 분류 과정에서 생성되는 RWKS 정보를 활용한다.

선행 연구([2][3][6][7])는 여러 개의 값을 읽을 때, 원장을 여러 번 주사(走査)해야만 했다. 이를

www.kci.go.kr

테면, 원장이 10번 블록까지 쌓인 경우, 키 A의 값이 들어있는 3번 블록과 키 B의 값이 들어있는 5번 블록을 읽기 위해서는 총 14블록(8블록+6블록)을 순차적으로 읽어서 A와 B를 발견해야 한다. 이는 다음 명령, 다음 트랜잭션에서 어떤 키를 요청할지 예측할 수 없으므로, A와 B를 한꺼번에 요청하지 못하고 필요할 때마다 값을 요청하기 때문이다.

하지만, RWKS 정보가 있는 경우, 하나의 트랜잭션 배치에 들어있는 모든 트랜잭션이 접근하는 키가 RWKS에 전부 들어있기 때문에, One-Pass Prefetching 기술이 적용된 트랜잭션 실행기는 스마트 컨트랙트가 필요로 하는 전체 값을 읽기 위해서 원장을 한 번만 주사하면 된다. 즉, 원장을 읽는 프로세스는 최신 블록에서부터 RWKS에 들어있는 모든 키를 검색, 발견할 때마다 값을 Global Cache에 적재하고, 해당 키를 RWKS에서 삭제한다.

트랜잭션 실행기는 필요한 값이 Global Cache에 있는지만 확인하면 되고, 없으면 필요한 키가 들어있는 블록을 아직 읽지 못한 것이니 기다리면 된다.

이를 통하여, 블록체인 실행 시간 중에서 가장 큰 시간을 차지하는 블록체인 원장 읽기 시간[10]을 최대 99% 이상, 전체 실행 시간을 최대 90% 이상 단축하였다.

5. TPAC 아키텍처의 성능 평가

실험에서는 데이터베이스의 성능을 평가하는 TPC-C 시나리오에 준거[9]하여, 저밀도·고밀도(트랜잭션 간 의존성) 및 시간적 국소성에 따른 성능을 측정하였다. 나아가, 데이터베이스와 블록체인의 기록 방식의 차이에 따른 차이를 분석하기 위해서, 참조 데이터의 기록 깊이에 따른 성능[9]을 분석하였다.

실험 대상은 소스 코드가 공개된 Hyperledger Fabric, NexLedger Accelerator, Ethereum[3], TPAC(제안 아키텍처)을 대상으로 분석하였다.

5.1 트랜잭션 간 의존성에 따른 성능 평가

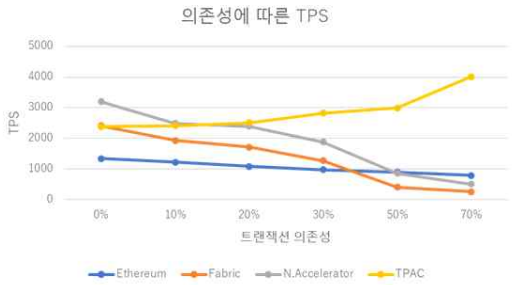

그림 3. 트랜잭션 의존성에 따른 TPS
Fig. 3. TPS based on transaction dependency

그림 3에서는 트랜잭션 간 의존성에 따른 성능 평가 결과를 보여주고 있다. 그래프로부터 볼 수 있듯이 트랜잭션 간 의존성이 높으면 높을수록 TPAC을 제외한 아키텍처의 성능이 저하되는 것을 볼 수 있다. 이는 트랜잭션 간 의존성이 높으면 높을수록 병렬처리에 의한 성능 향상 효과가 키 충돌로 인해 상쇄되기 때문이다. Hyperledger Fabric 및 NexLedger Accelerator의 경우 트랜잭션 의존성에 의한 성능 저하가 현저하다. Ethereum과 같은 단일 리더 노드 의존형 블록체인 아키텍처는 병렬처리에 의한 성능 향상이 없으므로 의존성에 의한 성능 저감 정도는 병렬처리 형 블록체인 아키텍처에 비해 낮다. 하지만, 의존도가 매우 높아지면 실질적으로 하나의 블록 안에 포함 가능한 트랜잭션 수가 제한되기 때문에 승인 가능한 트랜잭션이 줄어들어 성능 저하가 발생한다.

3) 공평한 비교를 위하여 Ethereum은 PoW 알고리즘이 아닌 PoA 합의 알고리즘을 사용하였음

www.kci.go.kr

5.2 시간적 국소성 요소를 고려한 트랜잭션 간 의존성에 따른 성능 평가

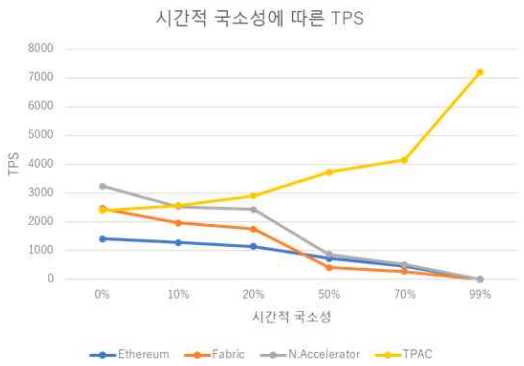

그림 4. 시간적 국소성에 따른 TPS
Fig. 4. TPS based on temporal locality

그림 4에서는 트랜잭션 간 의존성이 시간적 국소성에 영향을 받는 시나리오에 기반하여 성능을 평가하였다.

시간적 국소성에 영향을 받는다는 것은, 특정 키의 값에 대한 조회 또는 수정이 특정한 시간대에 집중된다는 뜻이다. 이를테면, 시간대에 따라 처리 요청 수가 일정하다고 하더라도, UTC 3:00 (한국표준시 12:00)에는 한국과 관련된 계정의 조회·수정 요청이 집중되고, UTC 17:00(미국 동부 표준시 12:00)에는 미국과 관련된 계정의 조회·수정 요청이 집중되는 등, 모든 키 값에 대한 조회·수정 요청 분포가 균일하지 않고, 시간대에 따라 편향될 수 있다. 즉, 특정 시간대의 트랜잭션만을 분석하면 트랜잭션 간 의존성이 더 높아질 수 있다. 모든 트랜잭션이 의존성을 가질 수 있는 사례로는 일련번호 발급이 필요한 요청 등이 있을 수 있다.

이러한 시나리오에서 성능 평가를 진행하는 경우, 매우 극단적인 성능 차이를 확인할 수 있다. Hyperledger Fabric, NexLedger Accelerator, 및 Ethereum은 매우 심각한 성능 저하를 보여준다. 이는 선행 연구[6]에서도 확인된 문제점이다. 트랜잭션 의존성이 높은 시나리오에서는 Hyperledger Fabric의 경우 0.8 TPS, NexLedger Accelerator의 경우 2.5 TPS 정도의 성능을 보여주고 있다. 이는 CPU나 메모리와 같은 환경과 무관한 시험 시나리오상에서 달성할 수 있는 이론상 최대치에 근접한 값으로, 키 충돌 문제의 중요성을 보여준다.

키 충돌 문제를 해결하지 못한 병렬처리형 블록체인에서는, 일정 시간(블록 생성 주기) 내에 수천수만의 트랜잭션을 실행, 결괏값을 생성하나, 그 모든 트랜잭션이 상호 의존성을 가지고 있으므로 일관성을 파괴하는 결과를 생성, 충돌로 인해 한 개를 제외한 모든 충돌 트랜잭션을 승인할 수 없다. 아무리 많은 트랜잭션을 실행하였어도 결과적으로 Hyperledger Fabric의 경우 블록당 1개, NexLedger Accelerator의 경우, 블록당 5개의 트랜잭션만 확정할 수 있다.

이에 반하여, TPAC과 같은 병렬처리형 블록체인에서는 시간적 국소성이 낮은 경우에 비해서는 성능이 저하되나, 일정 이상 낮아지지 않는 것을 확인할 수 있다. 이는 병렬처리형 블록체인에서는 의존성이 높아지면 병렬처리 가능 수는 줄어들지만, 한 개의 노드 그룹에서 처리할 수 있는 성능이 낮아지는 것이 아니기 때문이다. 오히려 의존성이 높으면 높을수록 캐시의 hit rate가 높아지기 때문에 단일 노드 그룹의 처리 성능은 향상된다.

5.3 원장 블록 깊이에 따른 성능 평가

그림 5에서는 참조 값이 기록된 블록의 깊이에 따른 성능을 비교한다. 그래프로부터 확인할 수 있듯이, 블록 깊이가 1000 이하면 성능상의 차이가 거의 없으나, 블록 깊이가 100,000 이상인 경우, TPAC 다른 모든 블록체인의 성능과 큰 차이를 보여주는 것을 확인할 수 있다. 100,000 이상

www.kci.go.kr

의 블록이 쌓이는 경우, 한 개의 값을 읽는 데에만 0.1~1sec 이상의 시간이 필요하므로, 트랜잭션 한 개의 실행 시간이 1초가 넘기 때문이다. 이는 I/O 처리 속도에 해당하기 때문에, 원장 파일구조에 대한 병렬 설계를 하지 않는 이상 해결할 수 없는 문제이다.

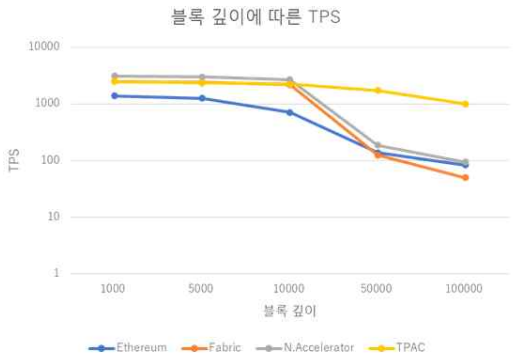

그림 5. 블록 깊이에 따른 TPS
Fig. 5. TPS based on block depth

6. 결 론

신기술을 산업에서 적용하는 경우, 안정성이 매우 중요하다. 블록체인의 경우, 기록에 대한 신뢰성 등 다양한 장점이 있다. 하지만, 5.1 및 5.2에서 보인 것과 같이 시나리오에 따라 TPS가 극단적으로 저하되는 등, 안정적인 성능을 보이지 못하는 문제로 인해, 성능적 안정성을 보장하지 못하면 실제 산업에 적용하기 어려운 문제를 가지고 있다. 또한, 블록체인으로 구축된 시스템이 지속해서 운용되는 경우 5.3과 같이 오래된 기록을 참조해야 하는 등의 경우 반응성 저하가 발생할 수 있다. 이는 IoT 등의 센서 기록을 보관하는 등의 블록체인 응용에서 고려해야 할 중요한 문제이다.

본 연구에서는 병렬 시뮬레이션을 위해 개발된 기술을 블록체인에 응용하여 이러한 문제를 해결한 블록체인 아키텍처를 설계·개발하였다. 이를 통해 실제 산업에서도 사용할 수 있는 수준의 안정적인 블록체인 기술을 확보하였다.

본 연구는 2021년도 정보통신기획평가원의 블록체인 융합기술개발 지원에 의한 연구임 [2020-0-00063]

참 고 문 헌

[1] 조용준, 김성보, 신동명, “분산처리 가능한 단일 원장 블록체인을 위한 Read/Write Key Set 정보 기반 트랜잭션 요청 분류 기법”, 2020년도 한국소프트웨어감정평가학회 추계 학술대회

[2] Amiri, Mohammad Javad, Divyakant Agrawal, Amr El Abbadi, “CAPER: A Cross-Application Permissioned Blockchain”, Proceedings of the VLDB Endowment 12, no. 11 (July 1, 2019): 1385 - 1398. DOI: https://doi.org/10.14778/3342263.3342275.

[3] Gorenflo, Christian, Lukasz Golab, Srinivasan Keshav, “XOX Fabric: A Hybrid Approach to Blockchain Transaction Executor”, In 2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), 1 - 9. Toronto, ON, Canada: IEEE, 2020. DOI: https://doi.org/10.1109/ICBC48266.2020.9169478.

[4] Y. Joe, S. Kim, D. Shin, “CACHED TRANSACTION EXECUTOR FOR DISTRIBUTED COMPUTING OF SINGLE LEDGER BLOKCHAIN”, The 7th International Conference on Electronics, Electrical Engineering, Computer Science. 2020.

[5] Y. Joe, K. Park, H. Kim, D. Shin, “One

www.kci.go.kr

Pass Value Prefetching for reducing read delay of blockchain", The 8th International Conference on Electronics, Electrical Engineering, Computer Science. 2021.

[6] Sharma, Ankur, Felix Martin Schuhknecht, Divya Agrawal, Jens Dittrich, "Blurring the Lines between Blockchains and Database Systems: The Case of Hyperledger Fabric", In Proceedings of the 2019 International Conference on Management of Data, 105–122. Amsterdam Netherlands: ACM, 2019. DOI: https://doi.org/10.1145/3299869.3319883.

[7] K. Lee, C. Yoon, K. Sung, N. N. Lincoln, K. Heo, R. Vaculin, R. Blessing-Hartley, A. O,Dowd, K. Um, "Accelerating Throughput in Permissioned Blockchain Networks", 2019.02 DOI: https://github.com/nexledger/accelerator/blob/master/docs/Whitepaper-Acceleratoring%20Throughput%20in%20Permissioned%20Blockchain%20Networks.pdf

[8] Androulaki, Elli, Artem Barger, Vita Bortnikov, Christian Cachin, Konstantinos Christidis, Angelo De Caro, David Enyeart, et al., "Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains", In Proceedings of the Thirteenth EuroSys Conference, 1–15. EuroSys '18. New York, NY, USA: Association for Computing Machinery, 2018. DOI: https://doi.org/10.1145/3190508.3190538.

[9] Difallah, Djellel Eddine, Andrew Pavlo, Carlo Curino, Philippe Cudre-Mauroux, "OLTP-Bench: An Extensible Testbed for Benchmarking Relational Databases", Proceedings of the VLDB Endowment 7, no. 4 (December 2013): 277–88. DOI: https://doi.org/10.14778/2732240.2732246.

[10] Miyamae, Takeshi, Takeo Honda, Masahisa Tamura, Motoyuki Kawaba, "Performance Improvement of the Consortium Blockchain for Financial Business Applications", Journal of Digital Banking 2, no. 4 (2018): 369–378.

[11] Sguanci, Cosimo, Roberto Spatafora, Andrea Mario Vergani, "Layer 2 Blockchain Scaling: A Survey", ArXiv:2107.10881 [Cs], July 22, 2021. DOI: http://arxiv.org/abs/2107.10881.

[12] Shavit, Nir, Dan Touitou, "Software transactional memory", Distributed Computing 10.2 (1997): 99-116.

[13] 시에 길루, 분산 원장 시스템에서 트랜잭션들의 병렬실행 수행 특허 출원번호 10-2019-7032260, 출원일 2019년 4월 12일, 등록일 2021년 8월 9일

[14] Bodon, F., L. Rónyai, "Trie: An Alternative Data Structure for Data Mining Algorithms", Mathematical and Computer Modelling, Hungarian Applied Mathematics, 38, no. 7 (October 1, 2003): 739–51. DOI: https://doi.org/10.1016/0895-7177(03)90058-6

www.kci.go.kr

저 자 소 개

조용준(YongJoon Joe)

2011.3 큐슈대학교 전기정보공학과 졸업
2013.3 큐슈대학교 정보학부 석사
2016.3 큐슈대학교 정보학부 박사과정 수료
2013.4-2016.3 일본 학술진흥원 특별연구원
2016.4-현재 : 엘에스웨어 수석연구원
<주관심분야> 병렬·분산 컴퓨팅, 게임이론, 분산 제약 최적화 문제

신동명(Dong-Myung Shin)

2003.2 대전대학교 컴퓨터공학과 박사
2001-2006 한국정보보호진흥원
응용기술팀 선임연구원
2006-2014 한국저작권위원회
저작권기술팀 팀장
2014-2016 한국스마트그리드사업단
보안인증팀 팀장
2016-현재 엘에스웨어(주)
연구소장/상무이사
<주관심분야> 오픈소스 라이선스, 시스템/네트워크보안, 저작권기술, SW취약점분석・감정, 블록체인 기술

www.kci.go.kr