The Transactions of the Korea Information Processing Society
TKIPS Vol.13, No.3, pp.102~110

eISSN: 3022-7011
https://doi.org/10.3745/TKIPS.2024.13.3.102

Reed-Solomon Encoded Block Storage in Key-value Store-based Blockchain Systems

Seong-Hyeon Lee† · Jinchun Choi†† · Myungcheol Lee†††

ABSTRACT

Blockchain records all transactions issued by users, which are then replicated, stored, and shared by participants of the blockchain network. Therefore, the capacity of the ledger stored by participants continues to increase as the blockchain network operates. In order to address this issue, research is being conducted on methods that enhance storage efficiency while ensuring that valid values are stored in the ledger even in the presence of device failures or malicious participants. One direction of research is applying techniques such as Reed-Solomon encoding to the storage of blockchain ledgers. In this paper, we apply Reed-Solomon encoding to the key-value store used for ledger storage in an open-source blockchain, and measure the storage efficiency and increasing computational overhead. Experimental results confirm that storage efficiency increased by 86% while the increase in CPU operations required for encoding was only about 2.7%.

Keywords : BFT, Blockchain, Erasure Code, Key-value Store, Reed-Solomon Encoding

키값 저장소 기반 블록체인 시스템에서 리드 솔로몬 부호화된 블록 저장

이 성 현† · 최 진 춘†† · 이 명 철†††

요 약

블록체인은 사용자가 수행하는 트랜잭션을 안전하게 기록 및 관리하기 위해 블록체인 네트워크의 참가자에 트랜잭션을 복제하여 저장하고 공유한다. 따라서, 블록체인 네트워크가 운영되는 동안 참가자들이 저장하는 전체 원장의 용량은 계속하여 증가하게 된다. 이러한 문제를 해결하기 위해 저장 효율성을 높이면서 참가자의 장치에 문제가 발생하거나 악의적인 참가자가 있는 경우에도 원장에 올바른 값을 저장할 수 있도록 보장해주는 방법의 연구가 진행되고 있다. 연구 중 한 방향은 리드 솔로몬 부호화와 같은 방식을 블록체인 원장 저장에 적용하는 것이다. 본 논문에서는 원장 저장을 위해 키값 저장소를 사용하는 오픈소스 블록체인에 리드 솔로몬 부호화를 적용하였고, 실험을 통해 이러한 부호화를 통해 얻을 수 있는 저장 효율성과, 증가하는 연산 오버헤드를 측정하였다. 실험 결과, 저장 효율성은 86% 증가하였으며 리드 솔로몬 부호화 과정에 필요한 CPU 연산의 증가 폭은 2.7% 정도로 적어서 부호화 방법의 유용성을 확인하였다.

키워드 : 비잔틴 장애 내성, 블록체인, 이레이저 코드, 키값 저장소, 리드 솔로몬 부호화

1. 서 론

블록체인 기술은 금융 분야를 넘어 다양한 영역에서도 활용되는 방안이 연구되고 있다. 예를 들어, 의료[1], 부동산[2], 공급망 관리[3] 등 여러 산업 분야에서 블록체인 기술이 도입되고 이를 통해 투명성, 신뢰성과 안전성을 높이면서 비용을 절감하는 방법이 제안되고 있다[4, 5]. 블록체인의 발전은 큰 변화를 가져오고 있지만, 그 과정에서 몇 가지 문제에 직면하고 있다. 비트코인의 경우, 전체 트랜잭션의 기록을 모두 저장하고 있는 풀 노드의 원장 크기가 2022년 1월 380GB에서 2023년 9월 491GB로 1년 반 만에 약 30% 증가하였다[6]. 앞으로 블록체인이 다양한 응용 분야에서 활용되기 시작하면 그 활용 분야에 따라 저장해야 할 원장 크기의 상승은 더 가속화될 것으로 예상된다. 블록체인 원장의 용량이 급증하면서, 블록체인 네트워크가 소수 마이닝 풀로 구성되는 중앙화의 위험

※ 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임(No. 2021-0-00136, 다양한 산업 분야 활용성 증대를 위한 대규모/대용량 블록체인 데이터 고확장성 분산 저장 기술 개발).
※ 이 논문은 2023년 ACK 2023의 우수논문으로 "오픈소스 블록체인 환경에서 리드 솔로몬 부호화 된 블록의 복구 성능 평가"의 제목으로 발표된 논문을 확장한 것임.
† 준 회 원 : 인제대학교 컴퓨터공학과 학사과정
†† 정 회 원 : 한국전자통신연구원 스마트데이터연구실 선임연구원
††† 종신회원 : 한국전자통신연구원 스마트데이터연구실 책임연구원
Manuscript Received : December 26, 2023
Accepted : January 15, 2024
* Corresponding Author : Myungcheol Lee(mclee@etri.re.kr)

성 문제가 제기되고 있고[7], 블록의 합의/저장 처리 속도 저하로 인해 트랜잭션의 검증 속도 또한 저하되는 등 많은 확장성 문제를 초래하는 블록체인 트릴레마 문제를 겪고 있다[8].

블록체인의 확장성 문제의 해결과 효율적인 네트워크 운영을 위해 샤딩(Sharding), 사이드체인(Sidechain), 라이트닝 네트워크(Lightning Network)와 같은 기술들이 개발되었다. 그 외에도 블록체인에 이레이저 코드(Erasure Code)를 적용하여 기존 복제 방식 대비 저장 부담을 많이 줄이는 방식이 연구되고 있다. 대표적으로는 Reed-Solomon(이하, RS) 부호화, Locally-Repairable Code(LRC) 부호화, Luby Transform(LT) 부호화, Raptor 부호화 등이 있다.

현재까지 많은 블록체인 시스템에서는 확장성과 성능 문제 해결을 위해 블록체인에 샤딩 방식을 접목하는 연구가 진행되고 있다. 블록체인 플랫폼 마다 각각 다른 특징을 갖고 있기 때문에 특정 블록체인에 맞는 새로운 샤딩 방식을 적용함으로써 얻는 이점에 대한 연구[9]와 샤딩 방식 적용에 따른 오버헤드, 네트워크 보안, 네트워크 연결 유지, 노드 동기화 등을 평가하는 연구를 통해 개선해야 할 문제점들을 파악하고 이를 해결하려는 연구가 진행되고 있다[10, 11].

샤딩 방식을 적용해 확장성과 성능을 높이는 실 사례로 이더리움이 있는데, 기존의 이더리움의 경우 낮은 TPS(Transaction Per Second)로 인해 거래 시 수수료가 매우 높다는 문제가 있다. TPS가 낮다는 것은 트랜잭션의 처리 속도가 오래 걸린다는 것인데, 이를 해결하려는 방법으로 이더리움은 샤딩 방식을 이용한 샤드 체인(Shard Chain)[12]의 구현을 고려하였다. 모든 노드가 트랜잭션을 저장하는 것이 아닌 샤드 체인을 도입해 각 노드가 일부의 트랜잭션을 보유함으로써 병렬처리 수행, 트랜잭션 분산으로 트랜잭션 처리 속도를 높여 확장성과 성능을 향상시려는 연구였다.

또한, 블록체인 분야에서 물리적 장애 또는 비잔틴 장애에 의해 발생하는 데이터 손실 오류를 RS 부호화를 적용하여 원장의 용량은 크게 줄이면서, 비잔틴 장애 내성(BFT: Byzantine Fault Tolerance)을 유지하도록 하는 연구를 수행하고 있다[13-16].

본 논문은 키값 저장소를 블록 저장소로 갖는 오픈소스 블록체인[17]에 RS 부호화를 적용하였고, RS 부호화 적용으로 인한 저장 효율성 및 블록체인 네트워크에서 피어 장애 또는 사용자 요구 때문에 발생하는 RS 부호화된 데이터의 복구 오버헤드를 성능 평가하는 연구를 수행하였다. 본 논문이 기여하는 바는 다음과 같다.

- 비트코인이나 하이퍼레저 패브릭 같이 파일 시스템의 블록 파일에 블록을 저장하는 방식이 아닌 키값 저장소 기반 블록 저장소를 갖는 오픈소스 블록체인 시스템에서 저장하는 블록, 즉 트랜잭션 데이터에 RS 부호화를 적용하여 비잔틴 장애 내성을 만족하면서 저장 효율성을 높이는 방법 제안
- 기반 블록체인 시스템과 RS 부호화 적용 시스템의 저장 효용성 및 CPU 사용률 비교를 통해 성능 평가 수행
- 노드 장애 발생 또는 비잔틴 노드 발견 시 복구 과정에 대한 실험 및 성능 평가 수행

2. 관련 연구

2.1 블록체인

블록체인은 탈중앙화된 분산 데이터베이스 기술로 데이터를 블록 단위로 저장하여 네트워크에 분산하여 저장한다. 각 블록은 이전 블록의 해시 등을 포함하여 순차적으로 연결되어 있다. 이러한 구조로 하나의 블록에 저장된 데이터 변경 시 연결된 모든 블록에 영향을 미치게 되어, 데이터 변경 및 조작을 아주 어렵게 하는 방법을 통해 데이터 무결성을 제공한다. 또한, 블록체인의 참여자들은 데이터 변경이나 추가를 합의하기 위해 합의 알고리즘을 사용한다. 아주 많은 선의의 참여자가 데이터를 검증하기 때문에 소수의 악의적 참여자가 데이터를 조작하기 어렵게 되어, 데이터 무결성이 유지되고 중앙 기관이 없이도, 블록체인에 기록된 데이터의 신뢰성이 보장되도록 한다.

비트코인 같은 경우 블록을 비트코인 참여자의 컴퓨터 속에 블록체인 형태로 저장한다[18]. 비트코인 네트워크 참여자는 지갑을 생성하고 블록을 다운로드하여 참여하게 되며, 새로운 블록이 추가되면 모든 참여 노드가 검증 후 추가된 블록을 체인에 연결하는 방식으로 동작한다.

하이퍼레저 패브릭 (Hyperledger Fabric)은 허가형 블록체인으로 블록체인 네트워크에서 발생하는 트랜잭션을 오더러(Orderer)에 의해 순서대로 모아 블록으로 저장, 관리한다[19]. 이때 저장되는 블록에는 블록의 번호와 현재 블록의 해시값, 그리고 이전 블록의 해시값을 보관하고 있다. 하나의 블록 파일의 최대 크기는 64MB이며, 블록과 관련된 설정값들은 오더러의 설정에 따른다.

기존 블록체인 네트워크에서 참여자들이 같은 내용의 트랜잭션을 공유하기 위해서는 저장한 블록 파일을 모든 참여자들이 같은 데이터를 복제하여 보관할 필요성이 있고, 이에 따라 같은 파일을 중복하여 보관하는 비효율적인 저장이 발생한다.

2.2 샤딩 기술

이더리움은 초당 수십 건의 거래조차 처리하기 어려워 여러 응용에서 활용되는 범용성에도 불구하고 더 이상 확장하기 어려운 확장성의 한계가 존재하였다[12]. 이를 해결하기 위해 샤딩 기술이 제안되었다. 샤딩 기술이란 하나의 큰 네트워크를 샤드라 불리는 여러 개의 작은 네트워크로 나누어 병렬적으로 동작하게 하는 기술이다[13]. 그러나, 샤딩 기술은 서로 다른 샤드에 포함된 계정 사이에서 교차 거래 시 샤드 동기화 등으로 인해 계산, 통신 비용이 더 많이 소모되는 문제가 있다[20]. 교차 거래를 최소화하기 위해 계정을 재배치하게 되면 반대로 샤드 별 계산 부하 격차가 발생하며, 이러한 문제를 해결하고자 하는 연구가 진행 중이다.

2.3 Blockchain_go

Blockchain_go는 본 논문에서 RS 부호화를 적용하고, 성능 평가를 진행하는 실험 환경으로 선정되었으며, Go 언어로 작성된 오픈소스 퍼블릭 블록체인 프로젝트이다[21]. Blockchain_go는 퍼블릭 블록체인인 비트코인과 유사하게 블록체인, UTXO(Unspent Transaction Output), 머클 트리, Mempool, 합의 과정 등이 구현되어 있다. 블록체인을 저장하는 저장소로는 파일 시스템을 이용하는 비트코인과 다르게 따로 서버가 필요하지 않은 임베디드 키값 저장소인 BoltDB[22]를 사용한다. 피어 간 통신은 가십 프로토콜과 같은 Peer-to- Peer 네트워크 대신 TCP/IP를 사용하는 특징을 갖고 있다.

Blockchain_go 시스템은 Fig. 1과 같이 모든 참여 노드들이 키값 저장소에 블록 번호와 직렬화된 블록을 키/값으로 동일하게 가지도록 구성되어 있다. 새로운 트랜잭션이 발생하는 경우, 블록이 생성된 후 같은 블록을 모든 노드가 중복 저장하여 저장 공간 낭비가 크다.

2.4 블록체인에 적용된 이레이저 코드

이레이저 코드(Erasure Code)는 데이터 저장 공간의 효율성을 높일 수 있도록 설계된 일종의 데이터 복제 방식이다. 이레이저 코드를 사용하여 데이터를 인코딩하고, 인코딩된 데이터의 일부가 손실되어도, 디코딩 과정을 거쳐 원본 데이터를 복구할 수 있도록 하는 기술이다.

가장 대표적인 이레이저 코드인 RS (Reed-Solomon) 코드는 통신에서는 낮은 에러율 환경에서 좋은 성능을 보이며, 이를 블록체인에 적용하면 노드 수가 고정적인 프라이빗 또는 허가형 블록체인 환경에서 좋은 성능을 보이는 부호화 방법이다. RS 코드는 (K, P) 형태로 나타낸다. K개의 데이터 심볼과 P개의 패리티 심볼을 더해 만들어진 총 N (= K + P)개의 데이터를 N 개의 노드/피어/저장소에 분산하여 저장하는 것을 의미한다. RS 코드는 P개까지의 에러가 발생해도 원본 데이터를 복원할 수 있다. 패리티 심볼의 개수(P)를 늘리면 에러 복구 성능을 증가시킬 수 있지만, 저장 공간 효율성은 떨어지며, 데이터 전송 시 전송량이 늘어나는 단점을 갖는다. 노드의 개수(N)에 의존하여 인코딩이 적용되고, 비잔틴 장애 내성의 보장 여부가 결정되기 때문에, 노드의 가입/탈퇴가 자유로운 퍼블릭 블록체인에 RS 코드를 적용한다면, 노드 수 변화에 따른 잦은 재인코딩이 CPU 및 네트워크 오버헤드를 초래할 수 있다.

이러한 RS 부호화 문제를 해결하기 위한 방안으로, Luby Transform(LT) 코드[23], Raptor 코드[24] 등의 심볼의 개수에 제약이 없는 무율(rateless) 부호화 방법을 적용하면, 블록체인 노드 수가 자주 변동되는 상황에서도 인코딩/디코딩을 자주 하지 않고도, 자원 효율적으로 블록체인 원장의 BFT 내성을 유지할 수 있다.

블록체인의 저장 효율성을 개선하기 위한 방법으로 데이터를 나누고 가상 노드를 추가하여 부하 분산과 중복 저장을 줄이는 연구[25]도 수행되었다. 다만 실행 시간의 증가와 블록체인 네트워크에서의 합의 도달의 어려움 등의 문제점이 존재한다. Raptor 부호를 이용하여 블록체인의 여러 블록들을 인코딩하는 방법에 대한 연구도 진행되었는데[26], 선형 복잡도의 부호화/복호화 방법을 제공하여 데이터의 가용성, 효율성을 보장하는 방법을 제안하였다. 하지만 부호화/복호화에 필요한 추가적인 자원이 필요할 수 있고, 따라서 대역폭이 낮거나 지연시간이 높은 블록체인 네트워크에는 적용하기 어려울 수 있다.

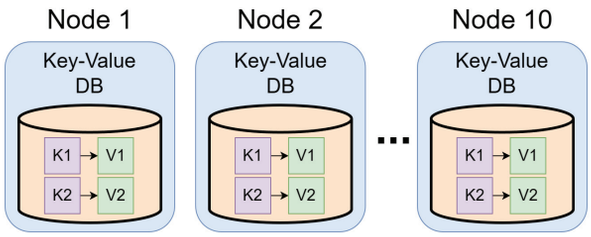

Fig. 1. Blockchain_go Ledger Replication Structure

3. 제안 시스템

3.1 시스템 구조

본 논문에서는 기반 시스템인 Blockchain_go의 원장에 RS 부호화를 적용하여 피어 및 비잔틴 장애에 의해 발생하는 데이터 손실 오류를 방지하면서 저장 용량을 줄이도록 개발하였다. 이렇게 부호화된 블록에 대해, 사용자가 원본 블록을 요구하거나, 피어의 장애 발생 시 블록을 제공하기 위해서는, 복구 과정, 즉 분산 저장된 인코딩된 블록을 전송받아서 재인코딩을 수행해야 하는 오버헤드가 있다.

Fig. 2는 본 논문에서 구현한 RS 인코딩이 포함된 오픈소스 블록체인의 시스템 구조를 나타낸다. Fig. 2는 본 논문의 시스템에서 블록 인코딩 과정(a1~a9)과 장애 복구 과정(b1~b5)을 각각 나타낸다.

블록 인코딩 과정은 먼저 블록체인 네트워크의 참여자가 거래를 진행하면(a1), 트랜잭션 생성자 (Transaction Generator)가 키값 저장소에 접근해 참여자의 주소, 잔액이 있는지 조회하여 트랜잭션이 올바른 트랜잭션인지 확인하는 트랜잭션 검증 과정을 수행한다. 검증된 트랜잭션은 서명을 하여 스토리지에 저장한다(a2, a3). 블록 생성자는 트랜잭션에 서명 등을 포함한 데이터를 모아서 블록을 생성한 후 DB에 저장한다(a4, a5). 블록 생성자는 이후 생성된 블록을 네트워크 내 다른 참여자에게 전송한다(a6). 그리고 각 참여자는 전송받은 블록들을 지역 DB에 저장하고, k개 이상의 블록이 모이면 RS 인코딩을 수행하여 데이터 샤드와 패리티 샤드를 생성한다(a7). RS 검증자 (RS Validator)를 통해 RS 인코딩이 잘 되었는

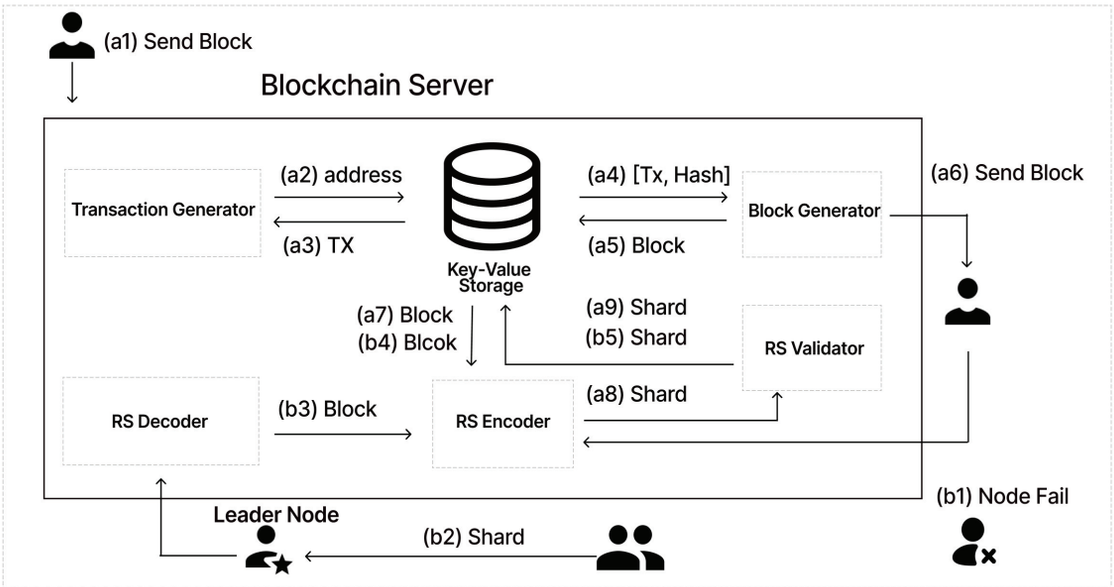

Fig. 2. Overview of Blockchain_go with RS Encoding

지 RS 인코딩 결과로 생성되는 샤드를 검증하여 인덱스를 생성하며(a8), 그리고 해당 참여자가 저장하도록 할당된 샤드만 DB에 저장(a9)하고 나머지 샤드는 버리는 과정을 포함한다.

장애 복구 과정은 샤드를 가진 참여자가 데이터를 손실하거나, 참여자가 네트워크에서 사라질 때(b1) 동작하는 과정을 각각 보여준다. 샤드를 가진 참여자의 데이터가 손실되면 해당 노드가 리더 노드가 되어 복구를 수행한다. 또는 사용자에게 특정 블록을 요청받은 노드가 해당 블록을 가진 참여자의 장애 또는 블록의 소실 상황을 파악하면 리더 노드로 선정되거나 다른 가용한 노드를 리더 노드로 선정하여 복구 과정을 시작한다.

리더 노드는 다른 노드들로부터 복구를 위해 필요한 청크를 전송받고(b2), RS 복호화를 통해 데이터를 복구한 후 복구된 블록을 모든 참여 노드들에 전파한다. 복구된 블록을 전송받은 노드들은 변화된 노드 환경에 맞게 재인코딩(b3, b4)을 수행하며, 이때 줄어든 참여자를 고려하여 비잔틴 장애 내성을 보장하도록 K와 P 값을 조정하여 재인코딩을 수행한다. 이후 모든 노드는 재인코딩 결과로 생성된 샤드들에 대해서 검증 과정(b5)을 수행하여 새로 인덱스를 구축한다. 마지막으로 검증된 샤드들은 다시 데이터 스토리지(DB)에 저장한다.

따라서, 본 논문에서 제안하는 시스템은 블록체인의 블록들에 RS 인코딩을 적용하여 블록을 데이터 샤드와 패리티 샤드로 나누고, 여러 노드에 분산 저장하여 가용성을 보장한다. RS 인코딩을 수행하는 기준은 블록 개수가 증가하면서 노드 수를 고려하여 일정 개수 단위로, 그리고 블록의 생명 주기를 고려하여 접근 빈도가 낮아지고 오래된 블록들을 우선적으로 선정하여 RS 인코딩을 수행한다. 이렇게 RS 인코딩이 적용된 블록들은 기존 전체 복제 방식과 대비하여 적은 공간을 차지하면서 여전히 BFT 내성을 보장할 수 있게 된다. 만일 RS 인코딩이 적용된 블록을 사용자가 요청하거나, 부호화된 데이터 또는 패리티 샤드를 가진 노드의 장애로 인하여 블록의 복구 가능성이 낮아지게 되면, 리더 노드를 선정하여 해당 블록에 대한 RS 디코딩을 선정하고, 재인코딩을 수행하여 해당 블록이 다시 BFT 내성을 보장할 수 있도록 한다.

RS 인코딩, 디코딩, 재인코딩 과정은 연산 및 데이터 전송 비용이 많이 발생할 수 있기 때문에, 특정 블록체인 시스템 및 환경에 적용할 때는 해당 비용이 어느 정도 발생하는지 실험을 통해 면밀하게 분석하여 시스템에 적용할 필요성이 있다.

3.2 설계 및 구현

본 논문은 기반 시스템에 블록 저장, 인코딩, 장애 발생, 샤드 전송, 디코딩, 재인코딩 등의 기능을 설계 및 개발하였고, 기반 시스템과 제안 시스템을 통해 RS 부호화 적용 전과 후의 블록 저장 효율성과 블록 복구 성능을 비교 평가하였다.

성능 평가를 위해, 전체 노드 수(N)의 1/3개까지의 장애(F)에 견딜 수 있는 비잔틴 장애 내성, 즉 $N = 3F + 1$ 공식을 만족하도록 $k = 7$, $p = 3$인 (7, 3) RS 부호화 환경을 구축하여 시험했다. 예를 들면, 10개의 데이터 중 3개까지의 데이터에 장애가 있을 때, $10 \geq 3F + 1$이기 때문에 비잔틴 장애 내성을 보장한다.

블록들을 저장하기 위한 스토리지는 키값 저장소인 BoltDB를 사용했다. 블록이 DB에 저장될 때 키와 값은 각각 블록 ID가 키로 사용되고, 블록 데이터는 바이트 배열로 변환하는 직렬화/역직렬화 과정을 통해 값으로 저장되고 조회된다.

성능 평가 환경에서 블록체인 네트워크에 참여하는 노드는 10개의 노드로 구성되며, 이때 비잔틴 장애 내성을 만족하려

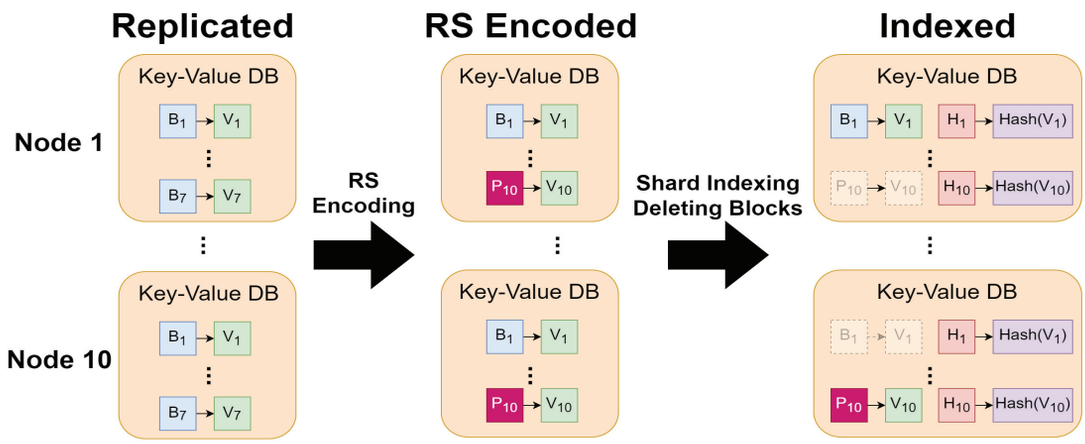

Fig. 3. Peer Ledger State Before/After Reed-Solomon Encoding

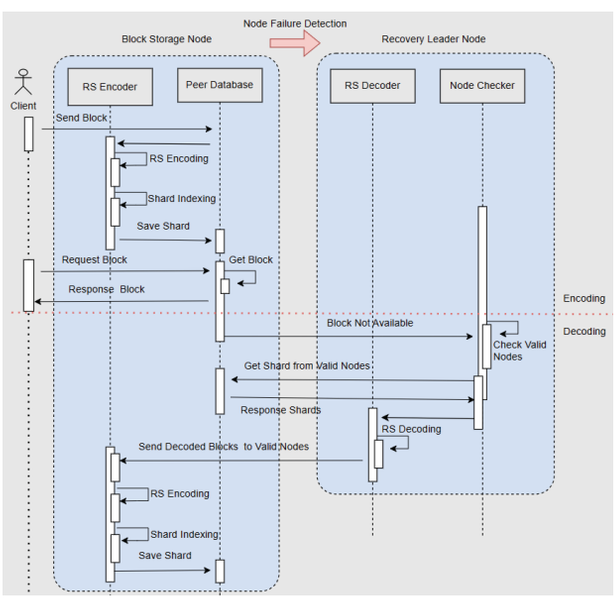

Fig. 4. Sequence Diagram for Block Encoding and Decoding

면, (7, 3) RS 인코딩을 수행해야 하며, 기본적으로 블록이 7개 생성될 때마다 RS 인코딩 라운드가 진행된다. (7, 3) RS 인코딩에 의해 7개 블록은 7개의 데이터 샤드와 3개의 패리티 샤드로 부호화된다.

Fig. 3과 Fig. 4는 원장 데이터인 블록들이 RS 부호화 및 인덱싱 과정을 거쳐 저장되는 과정을 나타낸다. Fig. 3에서 "Replicated" 상태는 RS 부호화가 적용되기 전에 참여 노드들의 원장 저장 상태, 즉 모든 노드가 모든 블록(B_1, …, B_7)을 키값 저장소에 중복하여 저장하는 상태를 나타낸다.

"RS Encoded" 상태는 각 노드가 저장한 블록에 (7, 3) RS 부호화를 적용하여 3개의 추가적인 패리티 블록을 생성한 상태를 나타낸다.

맨 마지막 "Indexed" 상태는 RS 부호화를 적용한 후, 각 노드가 담당하는 블록만을 저장하고, 담당하지 않는 노드는 삭제하여, 결국 원본 블록을 나타내는 샤드(B_1, …, B_7)와 패리티 블록을 나타내는 샤드 (P_8, P_9, P_{10})를 10개의 노드에 분산 저장한 상태를 나타낸다. 나중에 블록 복구가 필요하면 참여자들이 어떤 샤드를 나누어 가졌는지 알고 전송을 요청해야 하기 때문에, RS 인코딩된 전체 샤드를 인덱싱하여, 해당 샤드들의 해시 값이 인덱스 번호와 함께 참여자들의 DB에 저장되도록 한다. 이후 인코딩 및 인덱스가 완료된 샤드들은 참여자들의 원장 저장 DB에서 삭제한다.

블록 내에는 이전 블록의 해시, 현재 블록의 해시, 트랜잭션 등이 존재한다. 블록의 정보를 확인하기 위해서는 샤드를 저장하고 있는 블록체인 참여자들의 DB에서 바이트 배열로 구성된 데이터를 가져와 역직렬화 과정을 진행한 후 데이터를 반환하는 과정이 필요하다.

만약 블록 전송 요청을 처리하는 과정에 해당 블록의 장애 또는 통신 장애 등이 발생하여 해당 블록을 접근할 수 없다면 리더 노드를 선정하여 RS 복호화를 진행한다. 리더 노드는 복호화를 수행하기 이전에 어떤 블록이 복호화가 필요한지 기록하고, 복구에 필요한 블록들을 저장하고 있는 블록체인 참여자들에게 이전에 부호화한 데이터와 패리티를 요청한다. 디코딩에 필요한 데이터를 가지고 있는 블록체인 참여자들은 DB에서 요청한 데이터를 반환하고 전달받은 원본 블록, 패리티 블록들을 이용하여 복호화를 수행하여 장애가 발생한 블록을 복구한다.

블록체인의 참여자 중에 일부 참여자가 Fig. 4에서와 같이 악의적인 사용자에 의해 문제가 생기거나, 장치에 문제가 생겨서 비잔틴 장애 내성을 보장할 수 없는 경우, 새로운 노드 환경에 맞게 재인코딩을 수행하여 비잔틴 장애 내성을 보장할 수 있도록 한다. 장애가 발생한 노드에 저장된 데이터에 손실이 발생되었기 때문에, 재인코딩을 수행할 수 있는 리더 노드로 선정된 피어는 살아남은 다른 블록체인 참여자들의 DB에서 손실된 블록을 복구할 수 있는 정상 샤드를 k개 이상 가져와서 디코딩을 수행한다. 어떤 블록이 손실되었는지, 디코딩

시 필요한 데이터 샤드, 패리티 샤드 정보는 인덱싱 과정에서 생성한 인덱스로 알 수 있다. 데이터 샤드와 패리티 샤드를 포함해서 최소 k 개의 샤드를 전송받으면 RS 복호화를 수행하여 손실된 블록을 복구할 수 있다.

본 논문에서 재인코딩을 수행하는 리더 노드로 특정 노드를 선정하는 중앙 집중 방식을 채택하였으나, 탈중앙 방식을 적용하여 모든 노드에서 서로 필요한 데이터를 주고받아서 각각 재인코딩을 수행하는 방식을 선정할 수도 있다. 리더 노드는 다른 노드로부터 데이터 샤드와 패리티 샤드를 전송받아서 복호화를 진행하여 손실된 블록을 복구하여 다른 노드들에 전송한다. 이후 모든 노드는 새로 변경된 노드 환경에 맞게 재인코딩 과정을 진행한다.

블록체인 참여자가 이탈한 경우, 비잔틴 장애 내성을 만족하도록 노드 하나를 제외하고 인코딩한다. 10개의 노드 중 하나의 노드를 제외하고 인코딩해야 하므로 비잔틴 장애 내성 공식을 만족하도록 $9 \geq 3F + 1$, 즉 원본 샤드 7개와 패리티 샤드 2개를 가지도록 (7, 2) RS 인코딩을 수행한다.

만일 블록체인 참여자가 추가된 경우, 11개의 노드 상황에서 비잔틴 장애 내성 공식을 만족하기 위해서는 $11 \geq 3F + 1$, 즉 원본 샤드 8개, 패리티 샤드 3개를 가지도록 (8, 3) RS 인코딩을 수행하며, 인코딩된 데이터들은 이전과 같이 블록체인 참여자들 DB에 데이터 샤드, 패리티 샤드로 키값 저장소에 저장된다.

4. 성능 평가

본 논문에서 제안하는 키값 저장소 기반 블록 저장 방식을 갖는 블록체인 시스템에서 RS 부호화의 성능을 평가하기 위해 기반 시스템인 Blockchain_go의 소스 코드에 추가적인 기능을 구현하고, 다양한 실험을 수행하였다. 실험에 사용한 장비와 소프트웨어에 대한 정보는 Table 1과 같다. RS 부호화의 경우 (7, 3) RS 부호화를 사용하여 10개의 노드 중 최대 3개 노드의 오류에 대응할 수 있도록 하였다. 본 장에서는 실험을 통해 RS 부호화를 적용하는 경우 얻을 수 있는 저장 효용성과 비잔틴 장애 내성을 보장하면서 저장하는 데이터의 무결성을 보장하는 방법을 적용하는 경우 발생할 수 있는 오버헤드를 측정하였다. 먼저 기반 시스템과 RS 부호화를 적용한 시스템에서 블록 전송 시간의 비교를 수행하였다. 다음으로는 저장 효용성을 보이기 위해 기반 블록체인 시스템과 RS 부호화를 적용한 시스템에서 각각 키값 저장소에 저장하는 블록의 개수를 비교하였다. 그리고 각각의 경우 CPU 사용량을 측정하여 RS 부호화를 적용한 경우 발생하는 오버헤드를 측정하였다. 마지막으로 블록체인 네트워크에 참여하는 노드의 고장이나, 악의적 목적을 가진 비잔틴 노드의 참여를 가정하여, 부호화되어 분산 저장된 데이터를 복호화하고, 재인코딩하여 분배하는 상황을 가정하여 실험하고, 수행해야 하는 과정의 시간을 측정하였다.

Table 1. Experimental Environment

Environment	Value
CPU	Apple M1
RAM	16 GB
OS	Ventura 13.4
Programming Language	Golang (go 1.18.9)
RS Parameter (K)	7
RS Parameter (P)	3
Number of nodes	10
Number of blocks	7, 140, 280, 420, 560, 700

Fig. 5는 기반 블록체인 시스템과 RS 부호화를 적용한 경우 각각 블록 마이너가 모든 피어에 블록을 전송하는 데 걸리는 시간을 비교한 결과이다. 블록의 수에 따라 차이가 있지만, RS 부호화를 적용한 경우 17.65%에서 20.95%까지 블록 전송 시간이 증가하는 것을 볼 수 있다. 저장되는 블록의 개수가 증가할수록 참여 노드에서 블록을 저장하고 인코딩하고 검증하는 오버헤드에 의해 마이너로부터 블록을 전송받고 확인을 보내는 시간이 지연되는 것으로 보인다.

Fig. 6은 각 노드 수준에서 블록의 저장 공간을 비교한 결과이다. 기반 블록체인 시스템의 경우 모든 노드가 블록들을 중복 저장해야 하므로 10개의 노드가 모두 Fig. 6과 같이 각각 7개, 140개, 280개, 420개, 560개, 700개의 블록을 저장하여 총 70개, 1,400개, 2,800개, 4,200개, 5,600개, 7,000개의 블록을 저장하고 있으며, RS 부호화를 적용한 경우에는 패리티 샤드를 추가로 생성하고, 각 노드가 자신이 보관할 블록 또는 샤드를 인덱싱한 뒤 키값 저장소에 저장하게 된다. 이렇게

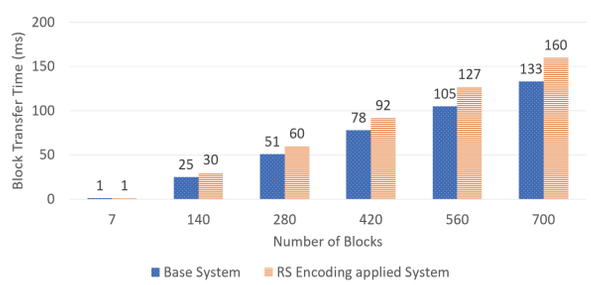

Fig. 5. Comparison of Block Transmission Performance

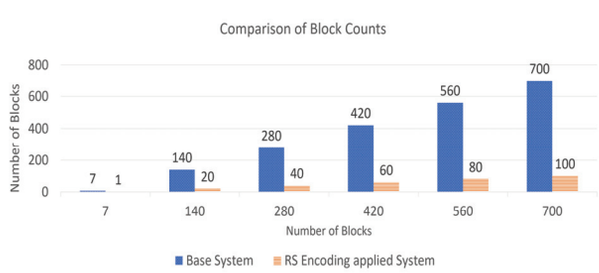

Fig. 6. Comparison of Block Storage Efficiency Per Node

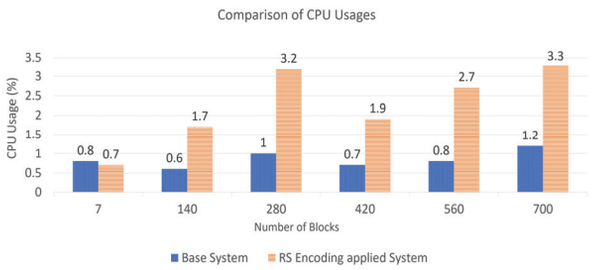

Fig. 7. Comparison of CPU Usage

하면 기존의 시스템에서 저장하는 공간보다 1/7의 공간만 사용하여 Fig. 6과 같이 각 노드당 1개, 20개, 40개, 60개, 80개, 100개 전체 10개 노드에는 총 7개, 140개, 280개, 420개, 560개, 700개의 블록을 저장하게 되어 공간 사용량을 1/7로 크게 줄일 수 있고, 7배 더 많은 블록을 저장할 수 있게 된다. 만일 노드 수가 더 많은 환경에서 RS 부호화를 적용하는 경우 k 값을 증가시켜서 공간 효율성을 더 높일 수 있다.

Fig. 7은 전체적인 블록의 증가에 따라 기반 블록체인 시스템과 RS 부호화를 적용한 경우 발생하는 CPU 사용량의 차이를 보여주는 결과이다. 먼저 기반 시스템에서도 트랜잭션 검증을 위한 동작을 수행하기 때문에 추가적인 CPU 사용이 발생하며, 실험 결과는 평균 0.8% 정도의 CPU 사용량이 발생하였다. 반면 RS 부호화를 적용한 시스템의 경우 기존의 시스템에서 수행하지 않았던 작업들, 즉 키값 저장소에 부호화를 한 블록을 저장하고 부호화 한 블록을 검증하는 과정 등이 포함되면서 CPU 연산을 추가로 수행하였다. 본 논문에서 제안하는 시스템의 구현을 통해 얻은 결과는 평균적으로 노드들이 3.5% 정도의 CPU 연산을 수행하여, 기반 시스템 대비 평균 2.7% 더 CPU 연산을 수행하는 것을 확인했다. RS 부호화 적용 시 가장 적게 CPU를 사용한 노드의 경우는 0.8%, 가장 많이 CPU를 사용한 노드의 경우는 8.1%까지도 사용하였다.

Table 2에서는 블록 복구 시간을 한 번에 복구하는 블록 개수에 따라 걸리는 시간, 그리고 복구 과정에서 세부 작업에 각각 걸리는 시간을 측정하였다. 또한, 복구 과정에서 복호화, 재부호화, 그리고 샤드 재전송 과정이 각각 전체 복구 시간에서 차지하는 비율을 계산하였다. 먼저 1개의 블록 복구의 경우, 복호화 과정은 0.017ms으로 전체 복구 과정에서 매우 적은 부분을 차지함을 볼 수 있었다. 재부호화와 샤드 재전송 시간이 각각 56%, 43%로 전체 복구 시간의 대부분을 차지하였다. 10개의 블록을 복구하는 경우, 복호화 시간은 1개 블록을 복구하는 경우와 큰 차이는 없었으나 샤드 재전송 시간이 증가하면서 전체 복구 시간의 79%를 차지하는 것을 확인할 수 있었다. 마지막으로 100개 블록의 복구 실험에서는 샤드 재전송의 시간이 전체 복구 시간에서 차지하는 비중이 조금 더 증가하였으며, RS 부호화를 적용할 때는 블록 복구에서 CPU 연산의 증가보다는 샤드 전송 비용을 줄이는 효율적 방법을 고안할 필요가 있음을 확인할 수 있었다.

5. 결 론

본 논문에서는 오픈소스 블록체인에 RS 부호화를 적용하여 얻을 수 있는 저장 공간 효율성과 비잔틴 장애 내성, 그리고 이 과정에서 발생하는 연산, 네트워크 오버헤드에 대한 성능 평가를 수행하였다. 실험을 통해 블록체인 네트워크에 참여하는 노드가 중복하여 저장되는 원장의 저장 공간을 RS 부호화를 통해 여러 노드에 분산 저장함으로써 약 86% 감소시킬 수 있음을 보였다. 또한, 참여한 노드에 문제가 발생하거나, 악의적인 목적을 가진 노드가 존재하는 경우 수행하는 복구 과정에서 디코딩, 재인코딩 등 CPU 연산 과정에서는 평균 3.5% 정도의 오버헤드, 기반 시스템 대비 평균 2.7% 정도 더 오버헤드가 발생하여 블록체인 네트워크의 동작에는 큰 영향이 없다는 것을 확인할 수 있었다. 그러나, 블록이 이미 많이 생성된 상태에서 복구를 위한 원본 샤드 및 패리티 샤드의 노드 간 전송은 여전히 큰 네트워크 비용이 발생할 수 있음을 실험을 통해 확인할 수 있었다.

Web 3.0의 시대로 도약하기 위해 블록체인 플랫폼에 대해서 확장성, 상호 운용성, 에너지 효율성을 향상시키기 위한 많은 연구가 진행되고 있다. 본 논문에서는 그중 하나인 Reed-Solomon 부호화를 이용해 블록체인 노드의 가용성 확보와 저장 공간 효율성 증가를 위한 방법을 키값 저장소를 원장의 저장소로 사용하는 오픈소스 블록체인에 적용하였고, 실험을 통해 성능 평가를 진행하였다.

실험 결과 저장 효율성을 크게 높여 확장성을 높일 수 있지만, 블록이 이미 많이 생성된 상태에서 복구를 위한 원본 샤드와 패리티 샤드의 노드 간 전송은 여전히 큰 네트워크 비용이 발생할 수 있음을 실험을 통해 확인할 수 있었다. 또한, 부호화 과정을 통한 블록 분산 저장 시 노드 간 동기화 여부 확인,

Table 2. Block Recovery Time by Number of Blocks

	Recover 1 Block		Recover 10 Blocks		Recover 100 Blocks	
	Time (ms)	Ratio (%)	Time (ms)	Ratio (%)	Time (ms)	Ratio (%)
Decode	0.017	0.06%	0.025	0.01%	0.048	0.02%
Re-encode	17.063	56.45%	40.674	21.13%	46.155	18.31%
Transfer	13.144	43.49%	151.762	78.85%	205.893	81.67%
Total	30.224	100.00%	192.461	100.00%	252.096	100.00%

네트워크 열결 및 안정성 유지, 다른 노드로부터 전송 받은 블록의 조작 가능성을 확인하는 방법의 제공 등 전체 부호화/복구 과정에서 상당히 세밀한 설계 및 구현이 필수적이다.

향후 연구에는 노드의 추가, 삭제, 장애에도 샤드 전송을 최소화할 수 있는 네트워크 프로토콜 개선 방안을 고민해 볼 수 있을 것이다.

References

[1] G. W. Hong and H. B. Chang, "A study on the design of medical service based on blockchain," *The Journal of Society for e-Business Studies*, Vol.28, No.1, pp.95-108, 2023.

[2] I. Park, "A study on the utilization of real estate block chain to the real estate transaction - Focusing on the propy case of international real estate transaction in the united states," Journal of the Korea Real Estate Management Review, Vol.21, pp.115-152, 2020.

[3] K. S. Oh and D. M. Lee, "The effect on the switching intention to the blockchain-based supply chain management information system," *Journal of Industrial Convergence*, Vol.20, No.12, pp.11-25, 2022.

[4] S. Lee, M. J. Park, N. H. Kim, and S. H. Seo, "Blockchain-based shared electric kickboard user management model," *KIPS Transactions on Computer and Communication Systems*, Vol.12, No.7, pp.217-226, 2023.

[5] M. J. Kang and M. H. Kim, "A hybrid blockchain-based e-voting system with BaaS," *KIPS Transactions on Computer and Communication Systems*, Vol.12, No.8, pp.253-262, 2023.

[6] BitInfoCharts, "Cryptocurrency statistics," Available: https://bitinfocharts.com/

[7] Trail of Bits, "Are blockchains decentralized?", Available: https://blog.trailofbits.com/2022/06/21/are-blockchains-decentralized/

[8] Q. Zhou, H. Huang, Z. Zheng, and J. Bian, "Solutions to scalability of blockchain: A survey," in *IEEE Access*, Vol.8, pp.16440-16455, 2020.

[9] A. A. Monrat, O. Schelén, and K. Andersson, "Addressing the performance of blockchain by discussing sharding techniques," *2023 3rd International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)*, Tenerife, Canary Islands, Spain, pp. 1-9, 2023.

[10] F. Hashim, K. Shuaib, and Zaki, N. "Sharding for scalable blockchain networks," *SN Computer Science*, Vol.4, No.1, pp.2, 2023.

[11] X. Liu, H. Xie, Z. Yan, and X. Laing, "A survey on blockchain sharding," *ISA Transactions*, Vol.141, pp.30-43. 2023.

[12] Web3 University, "Ethereum sharding: An introduction to blockchain sharding," Available: https://www.web3.university/article/ethereum-sharding-an-introduction-to-blockchain-sharding

[13] B. J. Choi, C. S. Kim, and M. C. Lee, "Research trends on distributed storage technology for blockchain transaction data," *Electronics and Telecommunications Trends*, Vol.37, No.3, pp.85-96, 2022.

[14] X. Qi, Z. Zhang, C. Jin, and A. Zhou, "BFT-Store: Storage partition for permissioned blockchain via erasure coding," in *Proceedings of the International Conference on Data Engineering (ICDE)*, pp.1926-1929, Apr. 2020.

[15] Y. Huang, M. Ye, and Y. Cai, "A Node Selection Scheme for Data Repair Using Erasure Code in Distributed Storage System," in Proceedings of the 6th International Conference on High Performance Compilation, Computing and Communications, in HP3C '22. New York, NY, USA: Association for Computing Machinery, Aug. 2022, pp.19-24. doi: 10.1145/3546000.3546003.

[16] G. Zhang et al., "Reaching Consensus in the Byzantine Empire: A Comprehensive Review of BFT Consensus Algorithms," ACM Comput. Surv., vol. 56, no. 5, p. 134:1-134:41, Jan. 2024, doi: 10.1145/3636553.

[17] S. H. Lee, and M. C. Lee, "Performance Evaluation of Reed-Solomon Encoded Block Recovery in Open Source Blockchain Environments," *Proceedings of the Annual Conference of Korea Information Processing Society Conference (KIPS) 2023*, pp.250-251, 2023.

[18] Satoshi Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System", https://bitcoin.org/en/bitcoin-paper

[19] Hyperledger Fabric, Available: https://www.hyperledger.org/projects/fabric

[20] D. H. Baek and S. W. Kim, "Adaptive load balancing algorithm of ethereum shard using bargaining solution," *KIPS Transactions on Computer and Communication Systems*, Vol.10, No.4, pp.93-100, 2021.

[21] blockchain_go, "Open Source Golang Blockchain Project," Available: https://github.com/Jeiwan/blockchain_go

[22] BoltDB, Available: https://github.com/boltdb/bolt

[23] S. Kadhe, J. Chung, and K. Ramchandran, "SeF: A secure fountain architecture for slashing storage costs in blockchains," *arXiv preprint, CoRR*, *arXiv:1906.12140*, 2019.

[24] A. Tiwari and V. Lalitha, "Secure raptor encoder and decoder for low storage blockchain," in *Proceedings of the International Conference on COMmunication Systems & NETworkS (COMSNETS)*, Bangalore, India, Jan. 2021.

[25] F. Meng, J. Li, J. Gao, J. Liu, J. Ru, and Y. Lu, "Blockchain storage method based on erasure code," in *2023 8th International Conference on Data Science in Cyberspace (DSC)*, Aug. 2023, pp.98-105. doi: 10.1109/DSC59305.2023.00024.

[26] D. Shi, X. Wang, M. Xu, L. Kou, and H. Cheng, "RESS: A reliable and efficient storage scheme for bitcoin blockchain based on raptor code," *Chinese Journal of Electronics*, Vol.32, No.3, pp.577-586, May 2023, doi: 10.23919/cje.2022.00.343.

이 성 현

https://orcid.org/0009-0006-8872-4661
e-mail : slee000220@oasis.inje.ac.kr
2019년 ~ 현 재 인제대학교 컴퓨터공학과 학사과정
관심분야 : Big Data, Blockchain

최 진 춘

https://orcid.org/0000-0002-6882-2890
e-mail : jcchoi@etri.re.kr
2011년 인하대학교 컴퓨터공학과(학사)
2013년 인하대학교 컴퓨터공학과(석사)
2020년 University of Central Florida, Computer Science(박사)
2022년 Texas A&M University-Kingsville, Electrical Engineering and Computer Science(박사후연구원)
2023년 ~ 현 재 한국전자통신연구원 스마트데이터연구실 선임연구원
관심분야 : Blockchain, Big Data, Information Security

이 명 철

https://orcid.org/0000-0002-1251-1727
e-mail : mclee@etri.re.kr
1999년 충남대학교 컴퓨터공학과(학사)
2001년 충남대학교 컴퓨터공학과(석사)
2001년 ~ 현 재 한국전자통신연구원 스마트데이터연구실 책임연구원
관심분야 : Big Data, Blockchain, Database, Distributed System