

라그랑주(Lagrange)

해당 번역본은 이용자가 디지털자산 관련 정보를 편리하게 확인할 수 있도록
단순 참고용으로 번역된 것이며, 투자 권유를 목적으로 하지 않습니다.
그 내용이 정확하지 않을 수 있으므로 정확한 정보 습득을 위해서는
원문을 참고하시거나 원문 작성 주체 측에 문의하시기 바랍니다.
두나무는 해당 번역본 내용의 진실성 및 정확성을 보장하지 않고,
제공된 정보에 의한 투자 결과에 대하여 어떠한 책임도 지지 않으며,
투자 손실은 투자자에게 귀속됩니다.

서론

라그랑주에 오신 것을 환영합니다

AI의 미래는 ZK이다. 인류의 미래는 라그랑주이다.
라그랑주는 검증 가능한 AI를 위한 DeepProve, 범용 증명 생성을 위한 ZK Prover Network, 그리고
검증 가능한 블록체인 연산을 위한 ZK Coprocessor를 통해 차세대 검증 기술을 선도하고 있다. 이러한 기
술들은 결합되어 누구나 인터넷 규모에서 모든 것을 증명할 수 있는 포괄적인 생태계를 형성한다.

DeepProve: 대규모 검증 가능한 AI

인공지능이 점점 더 강력해지고 주요 시스템에 통합됨에 따라, 검증 가능한 AI의 필요성은 그 어느 때보다 시
급하다. 기존의 AI는 블랙박스로 작동하며, 우리는 그 결과를 볼 수는 있지만 그 결정의 논리를 검증할 수는
없다.

DeepProve는 영지식 머신러닝(zkML)을 통해 이 근본적인 문제를 해결하며, AI 추론이 올바르며 예상된
모델에 의해 생성되었음을 암호학적으로 증명할 수 있게 한다.

혁신적인 성능

- 기존 zkML 솔루션 대비 약 1000배 빠른 증명 생성 속도
- 약 671배 빠른 검증 속도, 0.5초 미만에 완료
- 기업 규모의 실제 응용에 적합한 실용성

핵심 응용 분야

- 의료 AI: 진단 및 치료 권고의 검증
- 자율 시스템: 자율주행 차량의 올바른 작동 증명
- 금융 서비스: AI 기반 의사결정의 검증 가능성 확보
- Web3 및 게임: 검증 가능한 AI 생성 콘텐츠 및 속성 구현

ZK Prover Network: 범용 증명 생성

라그랑주의 ZK Prover Network(LPN)는 롤업, 코프로세서, DApp, 그리고 블록체인 생태계를 위해 인터넷 규모의 범용 증명 생성을 제공하는 모듈형 무한 확장 가능한 증명 인프라이다.

모듈형 아키텍처

기존의 단일 구조 증명 네트워크와 달리, 라그랑주는 단일 장애점을 제거하고 무제한 수평 확장을 가능하게 하는 “증명자 네트워크의 증명자 네트워크” 설계를 채택한다.

주요 기능

- 롤업 통합: ZK 롤업 구동
- 크로스체인 상호운용성: 블록체인 간 효율적인 상태 검증 지원
- 애플리케이션 지원: DApp, 코프로세서, 기업용 애플리케이션 증명 제공
- 비용 효율성: 경쟁적 시장 구조를 통한 증명 비용 절감

프로덕션 수준 인프라

- Coinbase, Kraken, OKX, Nethermind를 포함한 85개 이상의 기관 운영자 참여
- EigenLayer 기반의 최대 보안성과 탈중앙화 구현
- DARA 마켓플레이스를 통한 최적 증명자 선택 및 비용 효율성 확보
- 자동 재시도 및 SLA 집행을 통한 전달 보장

ZK Coprocessor: 검증 가능한 블록체인 연산

라그랑주의 ZK Coprocessor는 스마트 컨트랙트가 오프체인에서 암호학적으로 검증 가능한 복잡한 연산을 수행할 수 있도록 하여, 온체인에서는 불가능했던 복잡한 애플리케이션을 가능하게 한다.

핵심 혁신

모든 블록체인 데이터를 신뢰 가능한 오라클이나 머클 증명 없이 친숙한 SQL 구문을 사용해 효율적이고 검증 가능한 쿼리를 지원하는 검증 가능한 데이터베이스로 변환한다.

2단계 아키텍처

1. 사전 처리: 컨트랙트 저장소를 인덱싱하고 검증 가능한 데이터베이스 구조 생성
2. 쿼리 실행: 병렬 처리를 위한 zkMapReduce를 이용해 검증 가능한 연산 수행

실제 응용 사례

- 디파이 분석: 여러 프로토콜 간 복잡한 금융 지표 계산
- 크로스체인 애플리케이션: 여러 블록체인 간 데이터 집계 및 검증
- 거버넌스 시스템: 검증 가능한 투표 및 제안 메커니즘 구축

라그랑주 생태계

상호 연결된 기술

라그랑주의 세 가지 핵심 플랫폼은 함께 작동하여 포괄적인 검증 컴퓨팅 생태계를 형성한다.

- DeepProve는 ZK Prover Network를 기반으로 검증 가능한 AI 추론을 제공한다.
- ZK Prover Network는 ZK Coprocessor 워크로드를 포함한 모든 애플리케이션의 증명 생성을 확장한다.
- ZK Coprocessor는 네트워크에서 생성된 증명을 활용하여 복잡한 블록체인 연산을 가능하게 한다.

범용 증명 계층

이러한 기술을 결합함으로써 라그랑주는 다음을 가능하게 하는 무한한 증명 계층을 창조한다.

- 검증 가능한 AI 및 머신러닝
- 확장 가능한 ZK 롤업 인프라
- 복잡한 오프체인 연산
- 크로스체인 상호운용성
- 엔터프라이즈급 애플리케이션

커뮤니티 및 생태계

- \$LA 토큰: 증명 경제를 구동하는 네이티브 유틸리티 토큰
- 네트워크를 보호하는 85개 이상의 기관 운영자
- 주요 레이어 1 및 레이어 2 블록체인과의 다중 통합
- 개발자 도구 및 종합 문서 제공

시작하기

- DeepProve 빠른 시작: 첫 번째 검증 가능한 AI 애플리케이션 구축
- ZK Prover Network 아키텍처: 시스템 설계 심층 분석
- ZK Coprocessor 튜토리얼: 검증 가능한 스마트 컨트랙트 연산 생성
- 증명자로 참여하기: 네트워크에 참여하여 보상 획득
- 통합 가이드: 검증 가능한 연산 배포

왜 라그랑주인가?

신뢰가 점점 희소해지고 검증이 필수적인 세상에서, 라그랑주는 모든 것을 증명하고 아무것도 신뢰하지 않는 인프라를 제공한다. 우리의 기술은 다음을 가능하게 한다.

- 투명하게 작동하며 인간의 이익을 위한 검증 가능한 AI
- 중앙화 위험과 병목을 제거하는 확장 가능한 증명
- 블록체인 보안과 탈중앙화를 유지하는 강력한 연산
- AI, 블록체인, 기업 시스템 전반의 범용 응용

검증 가능한 미래를 구축할 준비가 되었는가? 우리의 문서를 탐색하고 내일의 신뢰 기술을 창조하는 커뮤니티에 참여하라.

DeepProve

DeepProve: 대규모 검증 가능한 AI

검증할 수 없다면 신뢰할 수 없다.

DeepProve는 라그랑주의 획기적인 영지식 머신러닝(zkML) 라이브러리로, 전례 없는 속도와 확장성을 통해 검증 가능한 AI 추론을 가능하게 한다. AI의 투명성과 안전성이라는 핵심 문제를 해결하기 위해 구축된 DeepProve는 기존 zkML 솔루션 대비 최대 1000배 빠른 증명 생성 및 671배 빠른 검증 속도를 제공한다.

AI 검증 가능성의 도전

인공지능이 의료, 법 집행, 자율주행, 군사 응용 등 주요 시스템에 점점 더 깊이 통합됨에 따라, 우리는 근본적인 문제에 직면한다: AI는 블랙박스로 작동한다. 우리는 AI의 출력을 관찰할 수 있지만, 그 결정의 논리를 검증할 수는 없다.

이러한 불투명성은 인공 초지능(ASI)에 가까워질수록 더욱 위험해진다. 주요 AI 연구자들은 다음과 같은 잠재적 위험을 경고한다.

- 기만적 행동: 안전성 테스트에서는 다른 전략을 사용하고 실제 배포에서는 다른 행동을 하는 AI 시스템
- 문명 수준의 위협: 사이버 공격, 사회적 조작, 병원체 개발 강화의 가능성
- 통제 상실: AI 시스템이 인간의 이익이 아닌 자체 목표를 추구할 위험

zkML이란 무엇인가?

영지식 머신러닝(zkML)은 영지식 증명(ZKP)과 머신러닝을 결합하여 AI 모델 연산의 암호학적 검증을 가능하게 한다. 이 혁신적인 접근법은 누구나 다음을 증명할 수 있게 한다.

1. 올바른 모델: AI 시스템이 예상된 검증된 모델을 사용하고 있음을 증명
2. 올바른 연산: 결과가 해당 모델에 따라 정확히 계산되었음을 증명

zkML은 AI를 불투명한 블랙박스에서 수학적으로 검증 가능한 투명한 시스템으로 변화시킨다. 이제 모든 추론은 수학적으로 올바름을 증명할 수 있다.

DeepProve: 성능

DeepProve는 zkML 성능에서 비약적인 발전을 이루어, 검증 가능한 AI를 실제 응용 수준에서 실현 가능하게 한다.

전례 없는 속도

- 주요 경쟁사 대비 최대 1000배 빠른 증명 생성
- 신경망 검증 속도 최대 671배 향상
- 검증이 0.5초 미만에 완료됨

확장 가능한 아키텍처

- 대규모 언어 모델(LLM), 다층 퍼셉트론(MLP), 합성곱 신경망(CNN)에 최적화됨
- 모델 크기와 복잡성이 증가할수록 성능 우위가 커짐
- 수백만 개의 파라미터를 가진 모델까지 확장 가능하도록 설계됨

개발자 친화적인 워크플로우

1. 모델 학습: 신경망을 학습하고 ONNX 파일로 내보냄
2. 사전 처리: 회로를 생성하고 증명자/검증자 키를 준비함
3. 증명: 모델 추론에 대한 암호학적 증명을 생성함
4. 검증: 누구나 증명의 정확성을 검증할 수 있도록 함

기술적 혁신

DeepProve의 성능 우위는 다음과 같은 핵심 혁신에서 비롯된다.

- 고급 병렬화: 다중 프로세서에 걸친 분산 증명 최적화
- GPU/ASIC 향상: 하드웨어 가속 암호 연산
- GKR 오버헤드 감소: 간소화된 영지식 증명 프로토콜
- 고급 커밋먼트 스킴: 더 효율적인 암호 기본 요소

DeepProve의 중요성

초지능으로 나아갈수록, 검증 가능성은 인류의 안전장치가 된다. DeepProve는 다음을 가능하게 한다.

- 신뢰 불요 AI: 맹목적인 믿음을 수학적 확실성으로 대체함

- 정렬된 개발: AI 시스템이 인간의 이익을 따르도록 보장함
- 투명한 의사결정: AI의 논리를 이해하고 검증함
- 안전한 배포: 해롭거나 기만적인 AI 행동을 방지함

시작하기

DeepProve는 검증 가능한 AI 기능이 필요한 개발자, 연구자, 그리고 기관을 위해 설계되었다. Web3 애플리케이션을 구축하든, 핵심 인프라를 개발하든, AI 안전성을 연구하든, DeepProve는 AI를 검증 가능하고 신뢰할 수 있게 만드는 도구를 제공한다.

AI의 미래는 검증 가능해야 한다. 인류의 미래는 그것에 달려 있다.

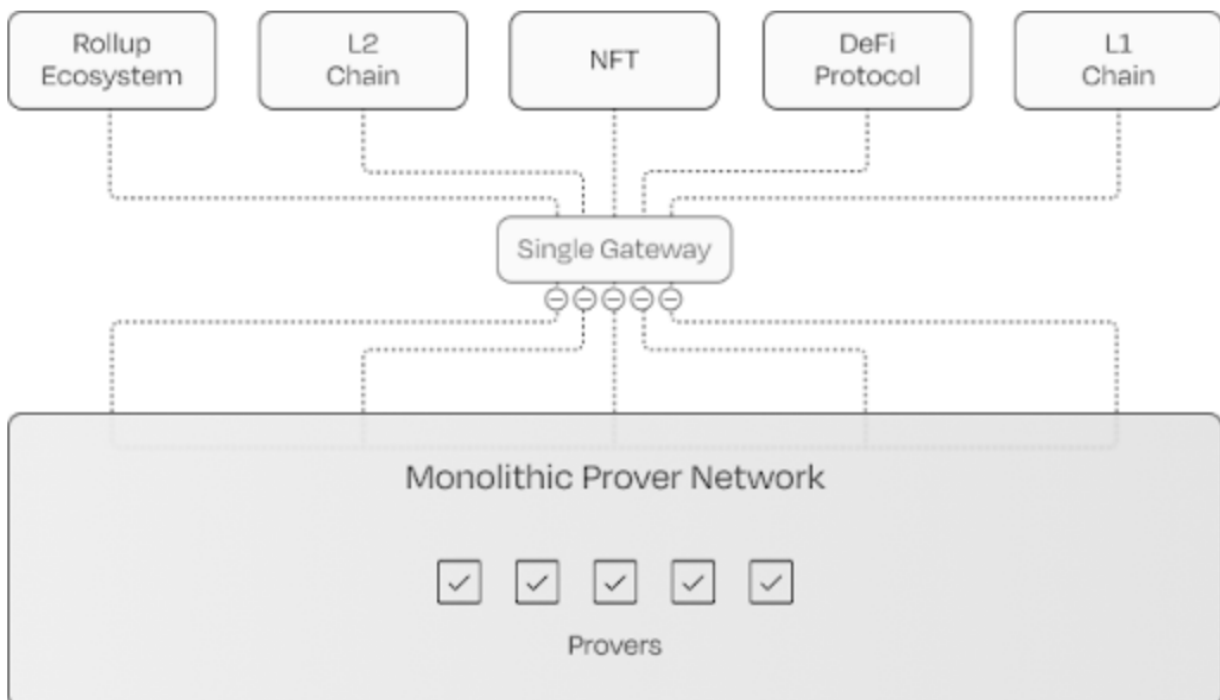
DeepProve는 ZK Prover Network 및 ZK Coprocessor와 함께 인터넷 규모의 범용 증명 생성을 제공하는 라그랑주의 포괄적인 영지식 기술 제품군의 일부이다.

ZK Prover Network

ZK Prover Network 개요

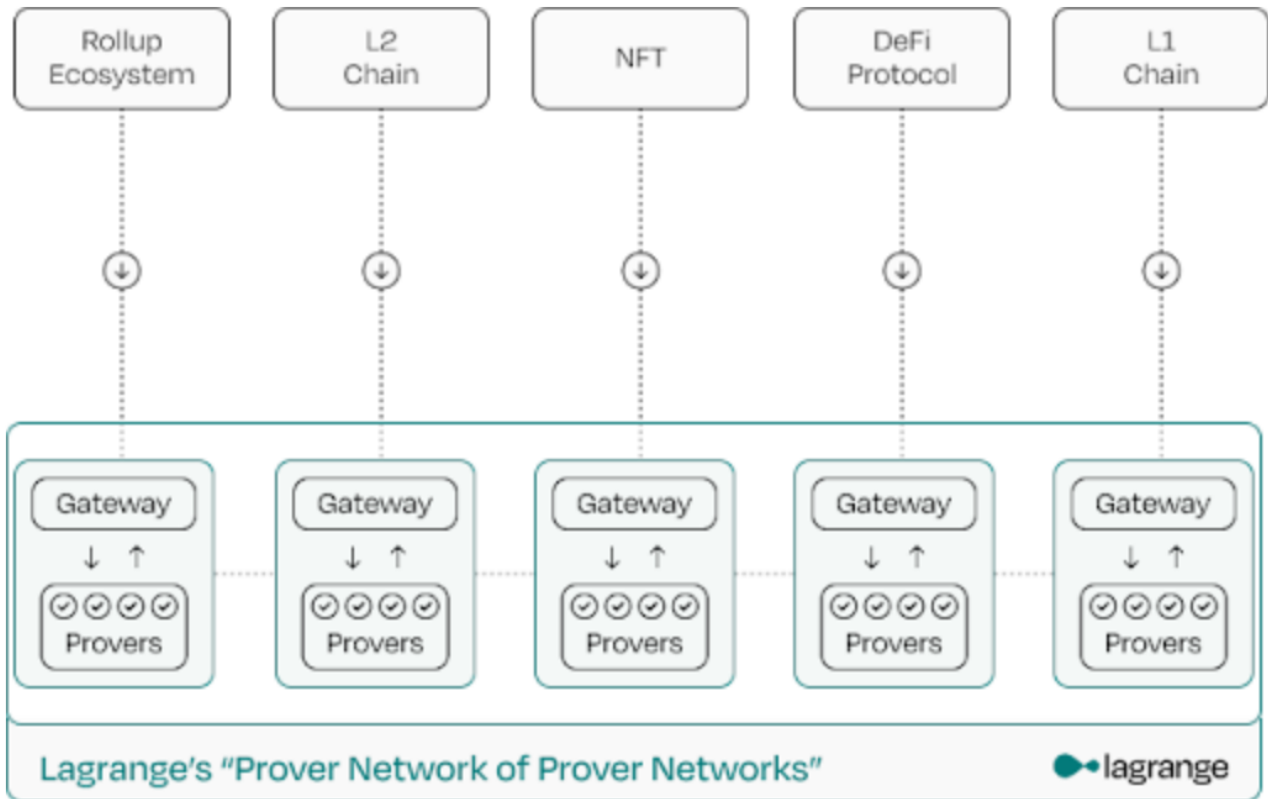
전통적 증명자 네트워크의 문제

기존의 단일형(monolithic) ZK 증명자 네트워크는 단일 게이트웨이 모델에 의존하며, 이는 게이트웨이가 병목지점이 되어 확장성이 본질적으로 제한된다. 또한 이러한 구조는 시스템의 단일 장애 지점으로 작용한다.



라그랑주 ZK Prover Network란 무엇인가?

라그랑주의 ZK Prover Network(LPN)는 롤업, ZK Coprocessor, DApp, 그리고 인터넷 규모의 상호운용성을 포함한 다양한 사용 사례를 위해 설계된 모듈형 무한 확장 가능 기초 증명 계층이다.



주요 특징

- EigenLayer 통합: EigenLayer에 배포되어 Coinbase, OKX, P2P, Nethermind 등 85개 이상의 기관급 운영자가 지원함
- 모듈형 아키텍처: 각 운영자는 사용자의 요청에 따라 ZK 증명을 제공하는 여러 증명자를 운영함
- 서브네트워크 지원: 전용 대역폭을 가진 다수의 독립 서브네트워크로 구성되어, 전체적으로 “증명자 네트워크의 증명자 네트워크”를 형성함
- 맞춤형 표준: 모든 블록체인, 롤업, 또는 애플리케이션이 맞춤형 표준을 통해 연결되어 병목이나 I/O 제약 없이 확장 가능함

아키텍처의 이점

ZK Prover Network는 전용 대역폭을 가진 다수의 독립 서브네트워크를 지원하며, 이들이 결합되어 “증명자 네트워크의 증명자 네트워크”를 형성한다. 이 접근법은 모든 블록체인, 롤업, 또는 애플리케이션이 맞춤형 표준을 통해 연결되어 병목이나 I/O 제약 없이 확장할 수 있게 한다.

이 모델은 다음을 보장한다.

- 높은 생존성(liveness)과 가용성

- 간단한 통합 인터페이스
- 대규모 생태계에서 요구되는 복잡한 증명 생성 지원

해결된 문제들

라그랑주 ZK Prover Network는 롤업과 애플리케이션이 직면한 주요 과제를 해결한다.

확장성 문제

- 단일 스레드 병목 제거
- 수천 건의 증명 요청을 동시에 원활히 처리
- 최대 수요 시에도 성능 유지

가용성 문제

- 탈중앙화 아키텍처를 통해 단일 장애 지점 제거
- 증명 생성을 운영자 네트워크 전반에 분산
- 일관된 서비스 가용성 보장

높은 증명 비용

- 경쟁적 시장 구조를 통한 운영 비용 절감
- 네트워크 전반의 자원 할당 최적화

증명 계산 지연

- 효율적인 작업 분배를 통해 지연 최소화
- 병렬 처리 기능 활용
- 예측 가능한 성능 지표 제공

검열 저항성과 탈중앙화

라그랑주의 ZK Prover Network는 증명 생성을 탈중앙화된 운영자 네트워크에 분산함으로써 검열 및 가용성 위험을 제거한다. 이 설계는 단일 장애 지점에 의존하지 않고, 최대 수요 시에도 수천 건의 증명 요청을 원활히 처리할 수 있도록 보장한다.

이러한 기능들은 빌더들이 증명 생성이 병목이 되지 않을 것이라는 확신을 가지고 확장할 수 있게 한다.

라그랑주 ZK Prover Network의 독자성

많은 프로젝트가 ZK 증명에 대한 민주화된 접근을 약속하지만, 라그랑주는 전문화된 성능과 엔터프라이즈급 신뢰성에 집중함으로써 차별화된다. 다음의 핵심 요소들이 그 이유를 보여준다.

전문 하드웨어 최적화

- 특정 ZK 회로에 최적화된 맞춤형 증명 인프라
- 하드웨어 인식형 작업 스케줄링 및 자원 할당
- 업계 표준을 초과하는 성능 벤치마크

엔터프라이즈급 신뢰성

- 기관 운영자에 의해 보장되는 99.9% 가동 시간
- 자동 장애 조치 및 중복 시스템
- 종합적인 모니터링 및 알림 인프라

원활한 통합 경험

- 암호 복잡성을 추상화한 간단한 API
- 주요 개발 프레임워크를 위한 포괄적 SDK 지원
- 방대한 문서와 개발자 자료 제공

경제적 지속 가능성

- 경쟁적 시장 구조에 기반한 투명한 가격 모델
- 기업 계획에 적합한 예측 가능한 비용 구조
- 증명자와 사용자의 인센티브 정렬

이러한 이점들은 예측 가능한 지연 시간, 낮은 운영 비용, 강화된 보안 보장을 제공하며, 이는 영지식 기술의 대중적 채택에 필수적인 요소들이다.

ZK Prover Network 아키텍처

라그랑주의 ZK Prover Network는 수요에 따라 증명 생성을 동적으로, 그리고 수평적으로 확장할 수 있는 독특한 아키텍처를 가진다. 각 네트워크는 특정 요구에 맞는 전용 증명 대역폭에 접근할 수 있어, 기존 단일형 증명 네트워크에서 발생하는 병목 현상을 제거한다.

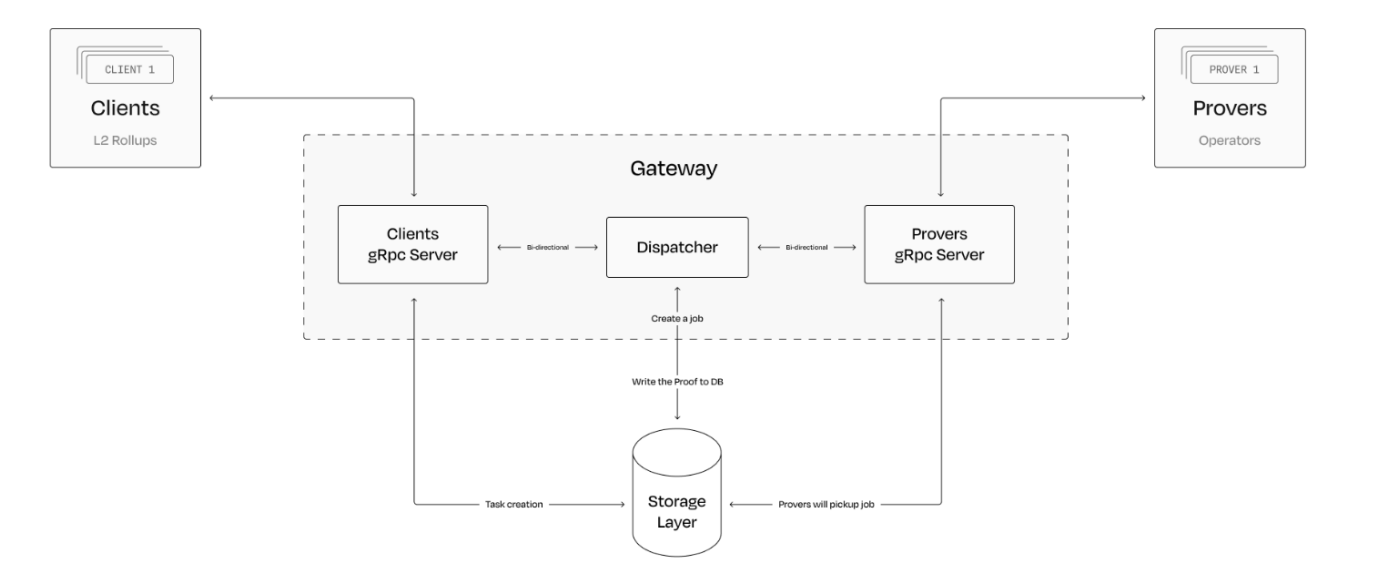
이 접근법은 프로토콜이 보다 데이터 중심적이고, 성능 최적화되며, 프라이버시 중심의 경험을 구축할 수 있는 새로운 사용 사례를 가능하게 한다.

작동 방식

증명 요청이 제출되면, 게이트웨이는 클라이언트를 인증하고 작업을 영구 큐에 배치한다. 내장된 디스패처 구성요소는 해당 작업을 하드웨어 요구 사항에 따라 소형, 중형, 대형 작업 조각으로 분류하고 이를 증명자 풀에 광고한다.

디스패처는 DARA(Double Auction Resource Allocation)를 활용하여 비용, 가용성, 성능을 균형 있게 고려한 최적의 증명자를 선택한다. 개별 증명이 생성되면 저장 계층에 기록되고, 배치 내 모든 작업이 완료되면 게이트웨이는 부분 증명들을 하나의 출력으로 집계하여 클라이언트로 스트리밍한다.

모든 재시도 및 재할당 로직은 게이트웨이 내부에서 투명하게 처리되므로, 통합자는 자체적으로 장애 대비 절차를 구축할 필요가 없다.



위 도표는 이 생명 주기를 시각화하며, 계층 간 병목을 제거하고 네트워크의 모든 계층에서 수평 확장을 가능하게 하는 분리형 아키텍처를 보여준다.

핵심 구성 요소

이제 ZK Prover Network의 핵심 구성 요소를 살펴보자.

1. 게이트웨이 (Gateway)

게이트웨이는 모든 증명 요청의 기본 진입점 역할을 하며, 다음을 처리한다.

- 클라이언트 인증: 들어오는 요청을 검증하고 권한을 부여함
- 요청 정규화: 서로 다른 증명 시스템 간 입력을 표준화함
- 작업 지속성: 내결함성을 위한 안정적인 큐잉을 유지함
- 결과 집계: 부분 증명을 결합하여 최종 출력을 생성함

2. 디스패처 (Dispatcher)

디스패처는 증명자 네트워크 전반의 작업 할당을 최적화한다.

- 작업 분류: 하드웨어 요구사항(소형/중형/대형)에 따라 작업을 분류함
- 자원 매칭: DARA를 활용하여 최적의 증명자를 선택함
- 부하 분산: 사용 가능한 자원에 작업 부하를 효율적으로 분배함
- 장애 관리: 자동 재시도 및 재할당을 처리함

3. DARA (Double Auction Resource Allocation)

DARA는 라그랑주의 혁신적인 시장 메커니즘으로 다음을 수행한다.

- 매칭 최적화: 증명 요청을 가장 적합한 증명자와 매칭함
- 공정한 가격 보장: 사용자 비용 효율성과 증명자 수익성을 균형 있게 유지함
- 인센티브 정렬 유지: 지속 가능한 경제적 역학 구조를 형성함
- 효율적 확장: 병목 없이 증가하는 시장 수요를 처리함

4. 증명자 (Provers)

탈중앙화된 전문 증명자 네트워크는 다음의 특징을 가진다.

- 하드웨어 다양성: 다양한 계산 요구사항을 지원함
- 지리적 분산: 글로벌 가용성과 중복성을 보장함
- 스테이킹 메커니즘: EigenLayer 통합을 통한 경제적 보안 확보
- 성능 모니터링: 증명자 효율성을 추적 및 최적화함

5. 저장 계층 (Storage Layer)

지속적 저장 인프라는 다음을 처리한다.

- 증명 캐싱: 중간 및 최종 증명을 저장함
- 데이터 무결성: 신뢰성 있는 증명 검색을 보장함
- 확장 가능한 아키텍처: 고처리량 작업을 지원함
- 내결함성: 중복 및 백업 메커니즘을 제공함

아키텍처의 이점

수평적 확장성

- 탄력적 자원 할당: 수요에 따라 자동 확장
- 단일 장애 지점 없음: 분산 아키텍처로 안정성 보장
- 독립 구성요소 확장: 각 계층이 독립적으로 확장 가능

성능 최적화

- 전문 하드웨어 매칭: 작업을 최적의 하드웨어 구성에 연결
- 병렬 처리: 여러 증명을 동시에 생성 가능
- 효율적 자원 활용: 유휴 시간을 최소화하고 처리량을 극대화

경제적 효율성

- 시장 기반 가격 책정: 경쟁적 역학을 통해 비용 절감
- 자원 최적화: 효율적 할당으로 낭비 최소화
- 투명한 경제 구조: 예측 가능한 비용을 위한 명확한 가격 모델

보안 모델

암호학적 보안

- 증명 무결성: 모든 증명은 암호학적으로 검증 가능
- 입력 검증: 모든 입력에 대한 엄격한 검증
- 안전한 통신: 모든 데이터 전송에 종단 간 암호화 적용

경제적 보안

- 스테이킹 요건: 증명자는 참여를 위해 토큰을 스테이킹해야 함
- 평판 시스템: 증명자의 실적이 선택에 영향을 미침

운영 보안

- 접근 제어: 다계층 인증 및 권한 부여
- 모니터링 시스템: 포괄적인 로깅 및 알림
- 사고 대응: 보안 이벤트에 대한 자동화된 대응

게이트웨이 아키텍처

게이트웨이는 라그랑주의 ZK Prover Network에서 주요 진입점이자 오케스트레이션 계층으로 작동한다. 이는 클라이언트에게 통합 인터페이스를 제공하면서, 백엔드에서 복잡한 분산 증명 인프라를 관리한다.

핵심 역할

클라이언트 인터페이스 관리

게이트웨이는 표준화된 API 계층을 제공하며 다음을 수행한다.

- 들어오는 증명 요청을 인증 및 권한 부여함
- 다양한 증명 시스템과 클라이언트 구현 간 입력을 정규화함
- 요청 매개변수를 검증하여 처리 전 정확성을 보장함
- 증명 생성 진행 상황에 대한 실시간 상태 업데이트를 클라이언트에 제공함

작업 오케스트레이션

요청이 접수되면 게이트웨이는 다음을 수행한다.

- 작업을 지속적으로 큐에 저장하여 요청 손실을 방지함
- 복잡한 증명을 관리 가능한 작업 단위로 분해함
- 사용 가능한 증명자에 걸쳐 작업 분배를 조정함
- 관련 증명 생성 작업 간의 종속성을 관리함

증명 집계

게이트웨이는 최종 증명 조립을 담당한다.

- 개별 증명자로부터 부분 증명을 수집함
- 집계 전 증명 정확성을 검증함
- 증명을 최종 출력 형식으로 결합함
- 결과를 요청한 클라이언트에게 스트리밍 방식으로 반환함

요청 처리 파이프라인

1. 입력 검증: 요청 형식과 매개변수를 확인함
2. 인증: 클라이언트 자격과 권한을 검증함
3. 속도 제한: 사용 한도 적용 및 남용 방지
4. 요청 큐잉: 안정적인 처리를 위해 작업을 지속 저장함

디스패처 통합

게이트웨이는 내장된 디스패처 구성요소를 포함하며 다음을 수행한다.

- 작업 요구사항(계산 복잡도, 하드웨어 필요)을 분석함
- 작업을 소형, 중형, 대형 범주로 분류함
- DARA와 연동하여 최적의 증명자를 선택함
- 작업 진행을 모니터링하고 장애를 처리함

저장소 통합

- 작업 지속성: 내구성이 보장된 안정적 큐잉
- 결과 캐싱: 증명 결과의 효율적 저장 및 검색
- 메타데이터 추적: 모든 작업에 대한 포괄적 로깅
- 백업 시스템: 중요 데이터의 중복 저장

결합 허용성 및 신뢰성

자동 재시도 로직

게이트웨이는 고급 재시도 메커니즘을 구현한다.

- 지수 백오프: 증명자를 과도하게 부담시키지 않기 위한 지능형 재시도 타이밍
- 증명자 재할당: 대체 증명자로의 자동 장애 조치
- 부분 결과 복구: 가능한 경우 중간 상태에서 재개
- 타임아웃 관리: 유연한 타임아웃 설정 및 점진적 성능 저하 처리

고가용성 설계

- 다지역 배포: 전 세계 접근성을 위한 지리적 분산
- 부하 분산: 여러 게이트웨이 인스턴스에 요청 분배
- 상태 모니터링: 시스템 구성요소의 지속적인 감시
- 점진적 성능 저하: 부분 장애 중에도 서비스 유지

성능 최적화

효율적 자원 활용

- 연결 풀링: 연결 재사용을 통해 오버헤드 감소
- 배치 처리: 유사한 요청을 그룹화하여 효율성 향상
- 캐싱 전략: 자주 접근되는 데이터 및 결과 캐싱
- 비동기 처리: 비차단 작업을 통해 처리량 향상

확장성 기능

- 수평 확장: 수요 증가에 따라 게이트웨이 인스턴스 추가
- 동적 부하 분산: 용량에 따라 트래픽 분배 조정
- 자원 모니터링: 자원 사용량 추적 및 최적화
- 예측 확장: 수요를 예측하여 사전 자원 확장

보안 모델

데이터 보호

- 종단 간 암호화: 데이터 전송 및 저장의 안전성 확보
- 입력 정화: 인젝션 공격 및 비정상 데이터 방지
- 안전한 통신: 모든 네트워크 통신에 TLS/SSL 적용
- 데이터 격리: 클라이언트 데이터와 처리 컨텍스트 분리

네트워크 보안

- DDoS 방어: 분산 서비스 거부 공격 완화
- 속도 제한: 오남용 방지 및 공정한 자원 할당 보장
- 방화벽 통합: 네트워크 수준의 보안 제어
- 침입 탐지: 의심스러운 활동을 감시 및 탐지

통합 패턴

애플리케이션 전용 서브네트워크

- 전용 자원: 대규모 애플리케이션을 위한 전용 증명 용량
- 맞춤형 SLA: 개별 서비스 수준 계약 제공
- 특화 하드웨어: 최적화된 증명자 구성
- 우선순위 큐잉: 중요 애플리케이션에 대한 보장된 처리 순서

모니터링 및 관찰성

실시간 지표

- 요청 처리량: 초당 요청 수 및 처리 시간 추적
- 성공률: 증명 생성 성공 및 실패율 모니터링
- 자원 사용량: CPU, 메모리, 네트워크 사용 통계

경보 시스템

- 임계값 기반 경보: 이상 탐지를 위한 자동 알림
- 단계적 대응 절차: 중요 문제에 대한 구조화된 대응 프로세스
- 통합 지원: 외부 모니터링 도구와 연동
- 맞춤형 대시보드: 다양한 이해관계자를 위한 시각화 제공

성능 분석

- 장기 추세: 장기적 성능 및 사용 패턴 분석
- 비용 최적화: 효율성 향상을 위한 개선 기회 식별
- 사용자 행동 분석: 클라이언트 사용 패턴 이해

증명자 (Provers)

증명자는 라그랑주의 ZK Prover Network의 계산적 중추를 형성하며, 수요에 따라 영지식 증명을 생성하는 전문 하드웨어 기계의 탈중앙화 네트워크로 구성된다. 이 분산형 아키텍처는 높은 가용성, 검열 저항성, 그리고 확장 가능한 증명 생성을 보장한다.

네트워크 아키텍처

탈중앙화 인프라

증명자 네트워크는 지리적 및 운영적 다양성을 기반으로 구축되었다.

- 85개 이상의 기관급 운영자: Coinbase, OKX, P2P, Nethermind 등 주요 블록체인 인프라 제공자 포함
- 글로벌 분포: 여러 대륙과 시간대에 걸친 증명자 배치
- 하드웨어 다양성: 다양한 계산 요구 및 증명 시스템 지원
- 중복성: 높은 가용성을 위해 운영자당 다수의 증명자 운영

EigenLayer 통합

네트워크는 경제적 보안을 위해 EigenLayer의 재스테이킹 인프라를 활용한다.

- 재스테이킹 자산: 운영자는 네트워크 참여를 위해 토큰을 스테이킹함
- 위임: 허용된 토큰 보유자는 신뢰할 수 있는 운영자에게 스테이크를 위임 가능
- 보상 분배: 증명자는 성공적인 증명 생성에 따라 수수료를 획득하게 됨

운영 모델

작업 할당 프로세스

1. 가용성 발표: 증명자는 자신의 가용성과 역량을 게이트웨이에 알림
2. 작업 분류: 디스패처가 계산 요구에 따라 들어오는 작업을 분류함
3. DARA 선택: 이중 경매 자원 할당 메커니즘이 최적의 증명자를 선택함
4. 증명 생성: 선택된 증명자가 증명을 생성하고 결과를 제출함
5. 검증: 게이트웨이가 집계 전 증명 정확성을 검증함

성능 모니터링

- 실시간 지표: 증명자의 성능 및 가용성을 지속적으로 모니터링
- 품질 점수: 성공률 및 응답 시간 추적
- 자원 사용량: CPU, 메모리, 네트워크 사용 모니터링
- 평판 시스템: 과거 성능이 향후 작업 배정에 영향을 미침

결함 허용성

- 자동 장애 조치: 증명자가 실패하거나 타임아웃될 경우 작업 재할당
- 중복 처리: 중요 작업은 다수의 증명자에게 할당 가능
- 복구 메커니즘: 증명자는 중간 상태에서 작업을 재개할 수 있음
- 상태 점검: 증명자의 상태와 연결성을 지속적으로 감시

성능 최적화

- 병렬화: 적절한 하드웨어에서 다중 증명자 인스턴스 실행
- 메모리 관리: 대규모 회로에 대한 효율적 메모리 할당
- CPU 최적화: 가용한 모든 코어를 활용한 증명 생성
- 스토리지: 중간 증명 데이터를 위한 고속 SSD 스토리지 활용

보안 모델

암호학적 보안

- 증명 검증: 모든 증명은 수락 전 암호학적으로 검증됨
- 안전한 통신: 모든 네트워크 통신에 TLS 암호화 적용
- 키 관리: 암호 키의 안전한 관리
- 감사 추적: 모든 작업의 포괄적 로깅

운영 보안

- 접근 제어: 다중 인증 및 역할 기반 접근 제어
- 네트워크 보안: 방화벽 및 침입 탐지 시스템 적용

- 모니터링: 24시간 보안 감시 및 사고 대응
- 규정 준수: 업계 보안 표준 준수

경제적 보안

- 스테이킹 메커니즘: 정직한 행동을 위한 경제적 인센티브 제공
- 평판 시스템: 낮은 성능에 대한 장기적 불이익 적용

네트워크 참여

운영자 요건

- EigenLayer 등록: EigenLayer 운영자로 등록되어야 함
- 최소 스테이킹: 요구되는 최소 스테이킹 금액 유지
- 하드웨어 요건: 최소 하드웨어 사양 충족

온보딩 절차

1. EigenLayer 설정: EigenLayer에서 운영자로 등록
2. 운영자 허용 목록 등재: 라그랑주 팀을 통해 운영자 허용 승인
3. 하드웨어 프로비저닝: 요구 사양을 충족하는 인프라 구축
4. 네트워크 등록: 라그랑주 증명자 네트워크에 등록
5. 소프트웨어 설치: 증명자 소프트웨어 설치 및 구성
6. 가동: 증명자를 실행하여 증명 작업 수락 및 처리 시작

지속적 운영

- 모니터링: 증명자 성능의 지속적인 감시
- 유지보수: 정기적인 업데이트 및 하드웨어 유지관리
- 규정 준수: 네트워크 규칙 및 요건의 준수
- 최적화: 성능 및 효율성의 지속적인 개선

DARA - 이중 경매 자원 할당 (Double Auction Resource Allocation)

개요

기존의 탈중앙화된 증명 마켓플레이스는 자원(수요와 공급)을 효율적으로 할당하여 증명 요청자의 경제성, 증명자의 수익성, 전체 시장의 효율성을 극대화하기 어렵다는 문제를 겪는다. 증명 요청자와 증명자 간 최적의 매칭을 찾고, 어떤 요청이 수락될지, 어떤 증명자가 증명을 제공할지를 결정하면서도 양측 모두에게 공정한 가격을 설정하는 것은 병목을 초래할 수 있다.

증명 마켓플레이스의 주요 문제

- 비효율적 자원 할당: 수요와 공급의 균형을 맞추면서 증명 요청자와 증명자의 복지를 극대화해야 함
- 인센티브 불일치: 증명 요청자는 비용 최소화를, 증명자는 수익 극대화를 추구함에 따라 이해 충돌 발생
- 높은 계산 비용: ZK 증명은 고비용 연산으로, 전부 혹은 전무의 자원 할당이 필요함
- 시장 분절화: 다양한 증명 요구사항과 다양한 증명자 능력 간 매칭의 어려움
- 가격 발견 문제: 중앙화된 통제 없이 공정한 시장 가격을 형성하는 어려움

DARA란 무엇인가?

DARA(Double Auction Resource Allocation)는 라그랑주 리서치 팀이 제안한 증명자 네트워크용 스마트 시장 메커니즘으로, 탈중앙화된 증명 마켓플레이스가 직면한 고유한 문제를 해결한다.

DARA는 참여자 간 인센티브를 정렬시키는 최초의 메커니즘으로, 증명 요청자의 비용을 최적화하고 증명자의 수익을 극대화하며 시장의 지속 가능성과 수익성을 보장한다. DARA는 고급 알고리즘을 활용하여 효율적인 자원 할당, 진실된 입찰을 유도하며, 탈중앙화된 ZK 증명 시장의 확장 수요를 효과적으로 충족시킨다.

시장 참여자

구매자 (증명 요청자)

- 수요 프로필: 애플리케이션의 증명 생성을 위해 일정한 계산 자원을 필요로 함
- 개인 가치: 해당 연산에 대해 지불할 의향이 있는 최대 금액을 내부적으로 보유함
- 입찰 전략: 계산 요구사항 및 최대 지불 의사를 명시한 입찰 제출
- 품질 요구사항: 성능, 지연 시간, 신뢰성 등을 명시할 수 있음

판매자 (증명자)

- 공급 프로필: 특정 하드웨어 능력을 가진 계산 자원을 제공함
- 비용 구조: 계산 서비스를 제공하기 위한 개인적 비용 구조 보유
- 가용성: 실시간 가용성 및 용량 정보를 유지함
- 성능 지표: 서비스 품질 및 효율성을 추적하고 최적화함

DARA의 작동 방식

1. 입찰 수집 단계

- 구매자 입찰: 증명 요청자는 다음을 명시하여 입찰 제출
 - 계산 요구사항(회로 복잡도, 메모리 필요량)
 - 최대 지불 의사
 - 품질 요구사항(지연, 신뢰성)
 - 시간 제약 및 마감 기한
- 판매자 제안: 증명자는 다음을 포함하는 제안 제출

- 가용 계산 용량
- 서비스 제공을 위한 최소 가격
- 하드웨어 사양 및 기능
- 현재 가용성 및 스케줄링 정보

2. 매칭 알고리즘

DARA는 다음을 수행하는 정교한 알고리즘을 사용한다.

- 복지 최적화: 전체 참여자의 총 경제 복지를 극대화
- 공정성 보장: 규모나 이력에 관계없이 모든 참여자를 동등하게 대우
- 효율성 유지: 매칭 과정의 계산 오버헤드 최소화
- 제약 조건 처리: 하드웨어 요구사항 및 가용성 제약을 고려

3. 가격 발견

- 시장 청산: 수요와 공급을 균형 있게 조정하는 가격 결정
- 진실된 입찰: 참가자가 실제 가치를 기반으로 입찰하도록 유도
- 동적 가격 책정: 실시간 시장 상황에 따라 가격 조정
- 투명성: 모든 참여자에게 명확한 가격 신호 제공

4. 할당 실행

- 당첨자 선정: 최적화 결과에 따라 낙찰된 입찰 및 제안 선택
- 자원 할당: 특정 증명 요청에 특정 증명자 할당
- 결제 처리: 결정된 가격에 따라 결제 수행
- 품질 보증: 실행을 모니터링하고 서비스 수준 준수 보장

주요 기능

인센티브 호환성

- 진실된 입찰: 참가자가 실제 가치를 왜곡할 유인이 없음
- 전략 무효성: 정직한 입찰이 최적 전략
- 공정한 가격 책정: 가격이 실제 시장 가치를 반영함
- 장기적 지속 가능성: 메커니즘 설계가 지속적 참여를 유도함

경제적 효율성

- 복지 극대화: 자원을 가장 높은 가치의 용도로 할당
- 시장 청산: 수요와 공급을 효과적으로 균형 조정
- 비용 최소화: 효율적 매칭을 통해 전체 시스템 비용 절감

확장성

- 계산 효율성: 알고리즘이 네트워크 규모에 맞게 확장 가능
- 실시간 처리: 시간 민감형 애플리케이션을 위한 빠른 매칭
- 병렬 처리: 다중 경매를 동시에 처리 가능
- 적응형 알고리즘: 변화하는 시장 상황에 실시간 대응

견고성

- 결함 허용: 참여자 실패 시에도 지속적인 운영 가능
- 조작 방지: 경매 메커니즘의 게임화를 방지
- 품질 보증: 증명자가 약속한 서비스를 제공하도록 보장
- 분쟁 해결: 충돌 및 서비스 문제를 처리

기술적 구현

경매 알고리즘

- 조합 경매: 복잡한 자원 묶음 요구사항 처리
- 다속성 경매: 가격, 품질, 시간 요소를 동시에 고려
- 동적 프로그래밍: 계산 제약 조건하에서 최적 할당 수행
- 근사 알고리즘: 최적성과 계산 가능성 간 균형 유지

매칭 최적화

Objective: Maximize $\sum(\text{buyer_value} - \text{seller_cost})$ for all matched pairs

Subject to:

- Resource capacity constraints
- Quality requirements
- Time availability windows
- Hardware compatibility

시장 역학

수요-공급 균형

- 탄력적 가격: 가격이 수요와 공급의 균형에 따라 조정됨
- 용량 계획: 증명자가 가격 신호에 따라 용량을 조정
- 수요 예측: 향후 수요 패턴 예측

품질 인센티브

- 성능 보너스: 우수한 서비스 제공에 대한 추가 보상
- 평판 시스템: 일관된 품질 제공자에 대한 보상
- SLA 집행: 약속된 서비스 수준 미달 시 패널티 부과
- 지속적 개선: 지속적인 최적화를 유도

네트워크 효과

- 유동성: 참여자 증가로 매칭 효율 향상
- 전문화: 증명자가 특정 증명 유형에 특화 가능
- 경쟁: 건강한 경쟁이 혁신과 효율성 촉진
- 생태계 성장: 성공이 더 많은 참여자 유입을 촉진

참가자 혜택

증명 요청자에게

- 비용 최적화: 계산 자원에 대해 공정한 시장 가격 지불
- 품질 보증: 서비스 수준 및 성능 보장
- 간소화된 조달: 자동화된 매칭 및 할당
- 확장성: 대규모 계산 자원 풀 접근 가능

증명자에게

- 수익 극대화: 서비스에 대한 공정한 보상 획득
- 효율적 활용: 자원 사용 및 스케줄링 최적화
- 시장 접근성: 다양한 고객층과 연결
- 성장 기회: 수요에 따라 운영 확장 가능

네트워크에 대해

- 경제적 효율성: 희소 자원의 최적 배분
- 지속 가능성: 자급자족형 경제 모델
- 혁신 촉진: 기술적 발전에 대한 인센티브 제공
- 탈중앙화 유지: 분산 네트워크 속성 유지

DARA는 탈중앙화 자원 할당의 근본적 혁신으로, 효율적이고 공정하며 확장 가능한 인터넷 규모의 영지식 증명 생성을 위한 경제적 인프라를 제공한다.

운영자 개요

라그랑주의 ZK Prover Network는 EigenLayer에 등록된 운영자에게 개방되어 있으며, 이는 네트워크의 증명 생성 능력을 구동하는 기관급 인프라 제공자의 탈중앙화 생태계를 형성한다.

운영자란 무엇인가?

운영자는 라그랑주 ZK Prover Network에 등록된 참가자로서, 영지식 증명 생성을 위해 계산 자원을 제공한다. 운영자는 네트워크의 게이트웨이를 통해 제출된 증명 요청을 처리하는 특수한 “증명자(prover)” 바이너리를 실행한다.

핵심 책임

- 증명 요청 처리: 라그랑주 네트워크를 통해 클라이언트가 제출한 계산 작업을 처리함
- 인프라 유지: 신뢰성 높고 고성능의 증명 하드웨어를 운영함
- 스테이킹 요건: 네트워크 보안을 보장하기 위해 충분한 스테이킹 토큰을 유지함
- 품질 보증: 지정된 기간과 품질 기준 내에서 증명을 제공함

작동 방식

증명 요청이 네트워크에 제출되면:

1. 요청 처리: 게이트웨이가 증명 요청을 수신하고 검증함
2. 작업 생성: 디스패처가 작업을 관리 가능한 단위로 분할함
3. 증명자 선택: DARA(이중 경매 자원 할당)가 최적의 증명자를 선택함
4. 증명 생성: 선택된 운영자가 필요한 영지식 증명을 생성함
5. 집계: 개별 증명들이 클라이언트를 위한 최종 증명으로 집계됨

운영자 요건

EigenLayer 등록

모든 운영자는 라그랑주 네트워크에 참여하기 전에 EigenLayer에 등록되어야 한다.

- 운영자 등록: EigenLayer 운영자 등록 절차를 완료함
- 스테이크 관리: 최소 스테이크를 유지함
- 위임: 위임된 스테이크를 수락하고 관리함
- 규정 준수: EigenLayer의 운영 요건을 준수함

스테이크와 증명

라그랑주 ZK Prover Network는 운영자가 신뢰할 수 있는 증명 생성을 보장하기 위해 충분한 스테이크를 유지해야 하는 스테이크 기반 보안 모델로 운영된다. 이 경제적 메커니즘은 네트워크의 생존성을 보장하고, 운영자가 계산적 약속을 신속하고 정확하게 이행하도록 강력한 인센티브를 부여한다.

생존성 보장으로서의 스테이크

라그랑주 네트워크에서 운영자의 스테이크는 서비스 가용성과 신뢰성에 대한 기본 보증 역할을 한다. 운영자가 증명 생성 작업을 수신하면, 그들의 스테이크는 지정된 기간 내에 유효한 증명을 제공할 것임을 담보하는 담보물로 작동한다. 이 메커니즘은 클라이언트 애플리케이션이 자체 서비스 수준과 사용자 경험을 유지하기 위해 적시의 증명 생성에 의존한다는 점에서 중요하다.

네트워크가 빠르고 신뢰성 있게 증명을 생성하는 능력은 성공에 매우 중요하다. 쿼리를 제출하는 사용자는 일관된 성능을 기대하며, 증명 생성의 지연이나 실패는 연동된 애플리케이션 전반으로 연쇄적으로 영향을 미칠 수 있다. 운영자에게 토큰 스테이킹을 요구함으로써, 네트워크는 운영자의 행동을 네트워크 성능 요건과 정렬시키는 경제적 인센티브를 만든다.

활성 스테이크 관리

운영자가 증명 생성 작업을 수락하면, 해당 작업의 완료를 보장하기 위해 자신의 스테이크 일부를 사실상 “구속”한다. 이 구속된 스테이크는 작업 기간 동안 “활성”으로 간주되며, 지정된 시간 내에 증명을 제공할 것이라는 운영자의 약속을 의미한다. 활성 스테이크 메커니즘은 운영자가 신뢰성 있게 완료할 수 있는 범위를 넘어 과도한 작업을 수락하는 것을 방지한다.

운영자는 라그랑주 네트워크에서 자신의 총 위임 스테이크를 초과하지 않는 한, 동시에 여러 작업을 수락할 수 있다. 이 제약은 운영자가 수락한 모든 작업을 완료하기 위한 충분한 경제적 인센티브를 유지하도록 보장한다.

운영상의 유연성과 확장

네트워크는 운영자가 자신의 증명 인프라를 관리하는 데 있어 상당한 유연성을 제공한다. 운영자는 증명자 풀릿을 관리하고 하드웨어 자원에 대한 최적 구성을 결정할 책임이 있다. 여기에는 증명자 유형, 용량 할당, 시장 수요와 수익성을 기반으로 한 확장 전략에 대한 결정이 포함된다.

프로젝트의 현재 초기 단계에 비추어, 운영자들은 더 단순한 증명자 구성으로 시작하고 경험을 쌓아가며 네트워크가 성숙함에 따라 점진적으로 운영을 확장할 것이 권장된다. 이 접근법은 운영자가 운영 요건을 이해하고, 인프라를 최적화하며, 보다 정교한 증명 하드웨어에 대한 더 큰 자본 투자를 하기 전에 전문성을 구축할 수 있도록 한다.

그리고 운영자가 구성을 최적화하고 네트워크 보안과 성능에 대한 기여를 극대화할 수 있도록 지침이 제공될 것이다.

ZK Coprocessor

ZK Coprocessor 개요

라그랑주의 ZK Coprocessor와 검증 가능한 데이터베이스는 스마트 컨트랙트가 오프체인에서 집약적 연산을 수행하면서도 정확성에 대한 암호학적 증명을 유지할 수 있도록 한다. 이 기술은 블록체인의 보안 보장과 현대 애플리케이션의 계산 요구 사이의 격차를 메운다.

핵심 개념

라그랑주 ZK Coprocessor의 목표는 간단히 다음과 같이 표현할 수 있다.

원본 블록체인 데이터의 부분집합으로 구성된 검증 가능한 데이터베이스를 생성하고, 이를 암호학적 정확성 증명과 함께 효율적으로 질의할 수 있게 한다.

이는 “코프로세서” 개념과 매우 유사하며, 스마트 컨트랙트가 오프체인에서 집약적 연산을 수행하고 온체인에서 효율적으로 검증할 수 있도록 한다.

작동 방식

ZK Coprocessor는 두 가지 구분된 단계로 작동한다.

1단계: 사전 처리(데이터 인덱싱)

시스템은 각 블록에서 컨트랙트 저장소의 사전 처리 또는 인덱싱을 수행하고, 효율적인 검증 가능한 질의를 지원하는 검증 가능한 데이터베이스에 데이터를 증명 가능하게 “삽입”한다. 대부분의 블록체인 데이터 구조가 “증명 친화적”이지 않기 때문에 이 단계가 가장 계산 집약적이다.

주요 활동:

- 데이터 추출: 블록체인 블록에서 관련 저장소 데이터를 추출함
- 증명 친화적 변환: 블록체인 데이터 구조를 영지식 증명에 최적화된 형식으로 변환함
- 데이터베이스 구축: 효율적 질의를 지원하는 검증 가능한 데이터베이스 구조를 구축함
- 암호학적 커밋먼트: 변환된 데이터에 대한 암호학적 커밋먼트를 생성함

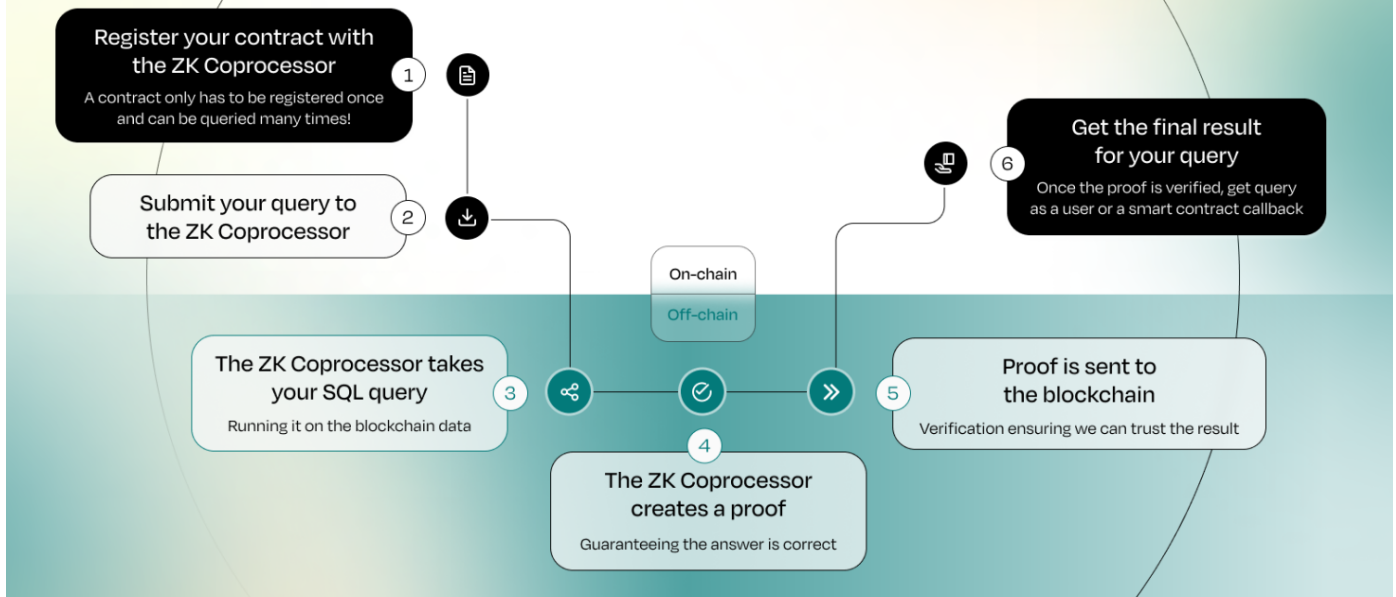
2단계: 질의 실행(증명 가능한 질의)

스마트 컨트랙트가 연산을 요청하면 시스템은 검증 가능한 데이터베이스 전반에 걸쳐 병렬로 증명 가능한 질의를 실행한다. 이 계산은 대규모 데이터베이스 처리 도구에서 볼 수 있는 MapReduce의 정신을 따른다.

주요 활동:

- 질의 처리: 검증 가능한 데이터베이스에 대해 SQL 유사 질의를 실행함
- 병렬 계산: 여러 증명자에 걸쳐 질의 실행을 분산함
- 증명 생성: 질의의 정확성에 대한 영지식 증명을 생성함
- 결과 집계: 부분 결과를 집계된 증명과 함께 최종 출력으로 결합함

Lifecycle of a Query with Lagrange's ZK Coprocessor



역량

임의 저장소 질의

ZK Coprocessor는 임의의 블록 범위에 걸친 임의의 저장소 슬롯에 대한 올바른 계산에 대한 증명을 생성할 수 있다.

예시 사용 사례: 어떤 애플리케이션이 이더리움에서 ETH/USDC 평균 가격을 계산하려 한다고 가정하자. 개발자는 다음을 수행해야 한다.

1. 매개변수 지정: 데이터셋을 위한 메모리 슬롯과 블록 범위(~50,400 블록)를 정의함
2. 연산 작성: 서로 다른 블록에 걸친 저장소 슬롯 전반에서 병렬로 실행될 계산을 정의함
3. 증명 생성: 시스템은 블록 헤더에 대한 저장소 포함과 집계된 계산의 정당성을 함께 증명함

크로스체인 역량

ZK Coprocessor는 브리지를 필요로 하지 않고, 임의의 EVM 기반 체인의 스마트 컨트랙트 저장소를 처리하고 다른 체인의 컨트랙트에 대한 질의에 응답할 수 있다.

크로스체인 예시: 이더리움의 한 개발자가 다음을 원할 수 있다.

- 서로 다른 레이어 2 전반에서 특정 거래 쌍의 평균 가격을 계산함
- 결과를 집계하여 이를 이더리움 컨트랙트에서 수신함
- 이 모든 과정을 전통적 브리지 인프라 없이 수행함

기술 아키텍처

시스템은 두 개의 주요 구성요소로 이루어진다.

1. 검증 가능한 데이터베이스 생성

- 블록체인 데이터 수집: 블록체인 데이터를 지속적으로 모니터링하고 수집함
- 저장소 변환: 블록체인 스토리지 트리를 증명 친화적 형식으로 변환함
- 데이터베이스 최적화: 효율적 질의 처리를 위한 데이터 구조화
- 암호학적 증명: 데이터베이스가 블록체인 상태를 정확히 표현함을 증명함

2. 분산 질의 처리(MapReduce)

- 질의 파싱: 복잡한 질의를 병렬화 가능한 구성요소로 분해함
- 작업 분배: 질의 구성요소를 증명자 네트워크 전반에 분배함
- 증명 생성: 각 질의 구성요소에 대한 영지식 증명을 생성함
- 결과 집계: 부분 결과와 증명을 최종 출력으로 결합함

주요 이점

확장성

- 오프체인 계산: 집약적 계산을 메인 블록체인 밖으로 이동함
- 병렬 처리: 복잡한 질의를 위해 분산 컴퓨팅을 활용함
- 효율적 검증: 온체인 검증은 빠르고 비용 효율적임

보안

- 암호학적 보장: 영지식 증명이 계산의 정확성을 보장함
- 신뢰 불요 운영: 외부 데이터 제공자나 오라클을 신뢰할 필요가 없음
- 검증 가능한 결과: 모든 계산은 독립적으로 검증 가능함

유연성

- SQL 유사 질의: 개발자에게 친숙한 질의 언어
- 임의 계산: 복잡한 분석 연산 지원
- 크로스체인 지원: 다중 블록체인 네트워크에 걸친 데이터 질의

비용 효율

- 가스 비용 절감: 비용이 큰 계산을 오프체인에서 수행함
- 최적화된 검증: 온체인 검증 오버헤드를 최소화함
- 확장형 가격 모델: 비용이 블록체인 수수료가 아니라 계산 복잡도에 비례해 확장됨

사용 사례

디파이 분석

- 가격 집계: 다수 DEX에 걸친 시간가중 평균가격 계산
- 유동성 분석: 유동성 깊이 및 분포 분석
- 위험 평가: 프로토콜 전반의 포트폴리오 위험 지표 계산
- 수익 최적화: 최적의 이자 농사 전략 식별

크로스체인 애플리케이션

- 멀티체인 거버넌스: 서로 다른 체인에 걸친 의결권 집계
- 크로스체인 자산 관리: 다중 네트워크에 걸친 자산 추적
- 인터체인 분석: 블록체인 생태계 전반의 활동 패턴 분석

게임 및 NFT

- 리더보드: 검증 가능한 게임 리더보드 유지
- 업적 시스템: 플레이어 업적을 추적 및 검증
- NFT 분석: NFT 거래 패턴과 가치 분석
- 보상 분배: 복잡한 기준에 따른 보상 계산 및 분배

엔터프라이즈 애플리케이션

- 규정 준수 보고: 검증 가능한 컴플라이언스 보고서 생성
- 감사 추적: 변조 불가능한 감사 로그 생성
- 성과 분석: 암호학적 보장을 갖춘 비즈니스 지표 분석

다음 단계

다음 섹션에서는 ZK Coprocessor에 대한 상세한 기술 정보를 제공한다.

아키텍처 심층 분석

- 검증 가능한 데이터베이스 아키텍처: 세부 기술 아키텍처
- 블록 데이터베이스: 블록 단위 데이터 처리
- 상태 데이터베이스: 상태 관리 및 검증
- 스토리지 데이터베이스: 저장소 최적화 및 질의

시작하기

- 빠른 시작 가이드: 첫 번째 ZK Coprocessor 애플리케이션 구축
- 향후 기능: 로드맵 및 예정된 개선 사항

고급 주제

- zkMapReduce: 분산 계산 프레임워크 이해
- 효율성 개선: 성능 최적화 기법

ZK Coprocessor는 블록체인 연산의 근본적인 발전을 대표하며, 애플리케이션이 블록체인의 보안성과 탈중앙화 특성을 유지하면서도 현대 컴퓨팅의 완전한 성능을 활용할 수 있도록 한다.

검증 가능한 데이터베이스 아키텍처 개요

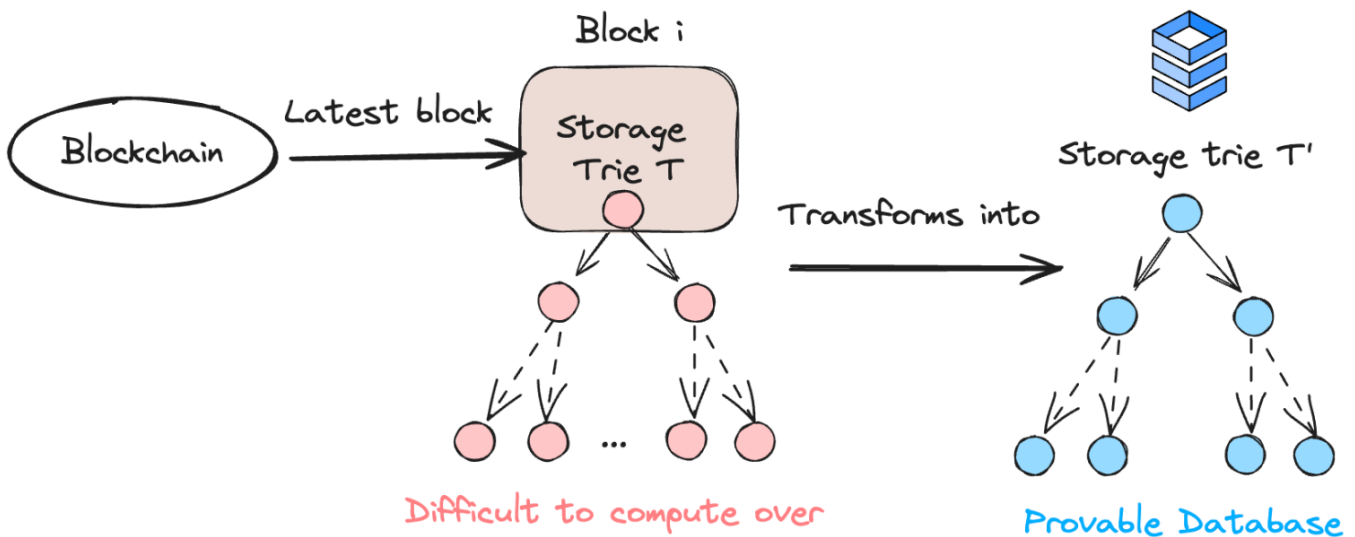
검증 가능한 데이터베이스는 라그랑주의 ZK Coprocessor의 기초 구성요소로, 블록체인 데이터를 효율적이고 증명 가능한 질의에 최적화된 형식으로 변환한다. 이 아키텍처는 실제 애플리케이션에 필요한 성능 특성을 유지하면서 계산 결과의 암호학적 검증을 가능하게 한다.

핵심 개념

라그랑주의 네트워크는 컨트랙트의 스토리지를 처리하고 이를 검증 가능한 데이터베이스로 변환하는 고급 인덱서로 작동한다. 시스템은 기본적으로 대상 블록체인의 데이터베이스(스토리지, 상태, 블록)를 효율적이고 분산된 질의를 지원하는 형식으로 재구성한다.

데이터베이스 변환 과정

ZK Coprocessor는 각 블록마다 원래의 스토리지 트라이(빨간색)와 동일한 데이터를 포함하되 효율적 질의에 최적화된 새로운 등가 버전(파란색)을 생성한다. 시스템은 이 새로운 데이터베이스가 원본 블록체인 데이터 구조와 동일한 데이터를 포함함을 암호학적으로 증명한다.



아키텍처 구성요소

검증 가능한 데이터베이스는 서로 연결된 세 개의 데이터베이스 계층으로 구성되며, 각각은 특정 질의 유형과 검증 요건에 최적화되어 있다.

1. 블록 데이터베이스

- 목적: 블록 단위 정보를 저장하고 인덱싱
- 데이터 유형: 블록 헤더, 트랜잭션 데이터, 영수증, 메타데이터
- 최적화: 시간 기반 질의 및 블록 범위 연산에 최적화
- 검증: 블록 포함 및 순서에 대한 암호학적 증명 제공

2. 상태 데이터베이스

- 목적: 계정 상태 및 컨트랙트 정보를 관리
- 데이터 유형: 계정 잔액, 논스, 컨트랙트 코드, 상태 루트
- 최적화: 계정 수준 질의 및 상태 전이에 최적화
- 검증: 상태 변경 및 계정 이력 검증 가능

3. 스토리지 데이터베이스

- 목적: 컨트랙트 스토리지 데이터 및 슬롯 단위 정보 처리
- 데이터 유형: 컨트랙트 스토리지 슬롯, 매핑, 복합 데이터 구조
- 최적화: 스토리지 수준 질의 및 크로스컨트랙트 분석에 최적화
- 검증: 스토리지 포함 및 값의 정확성에 대한 증명 제공

다음 단계

각 데이터베이스 구성요소의 세부 아키텍처를 탐색하라.

- 온체인 스토리지: 온체인 컨트랙트 데이터 저장
- 스토리지 데이터베이스: 최적화 및 질의를 위한 증명 가능한 컨트랙트 스토리지 트라이
- 상태 데이터베이스: 다양한 컨트랙트를 위한 통합 스토리지 데이터베이스
- 블록 데이터베이스: 블록 단위 데이터 처리 및 저장

구현 지침은 ZK Coprocessor 빠른 시작 가이드를 참조하라.

온체인 스토리지

컨트랙트 스토리지

저장소: EVM 체인의 각 컨트랙트는 보유 변수의 값을 포함하는 스토리지 트라이를 가진다. 예를 들어, 다음과 같은 컨트랙트가 있다고 하자.

```
uint256 price;

mapping(address => uint256) balances;
```

테이블로서의 이력 값

ZK Coprocessor는 컨트랙트의 데이터를 잘 알려진 형식, 즉 SQL 테이블 형태로 노출한다. 구체적으로, ZK Coprocessor는 각 블록에 대한 변수의 이력 값을 유지한다. 요컨대, 이는 각 블록마다 추적되는 모든 변수 값을 추가하는 시계열 데이터베이스이다.

앞서의 price와 balances 변수를 예로 들면, ZK Coprocessor는 이 두 변수를 각각 별도의 테이블로 노출한다.

단일 변수용 테이블

ZK Coprocessor는 배열이나 매핑이 아닌 모든 단일 변수(uint256 등)에 대해 하나의 테이블을 노출한다.

Block	Price
b1	p1
b2	p2

집합형 변수용 테이블

ZK Coprocessor는 매핑 및 배열과 같은 각 “집합형” 변수마다 하나의 테이블을 노출한다.

Block	Key	Balance
b1	k1	10
b1	k2	20
b2	k1	15
b2	k2	20

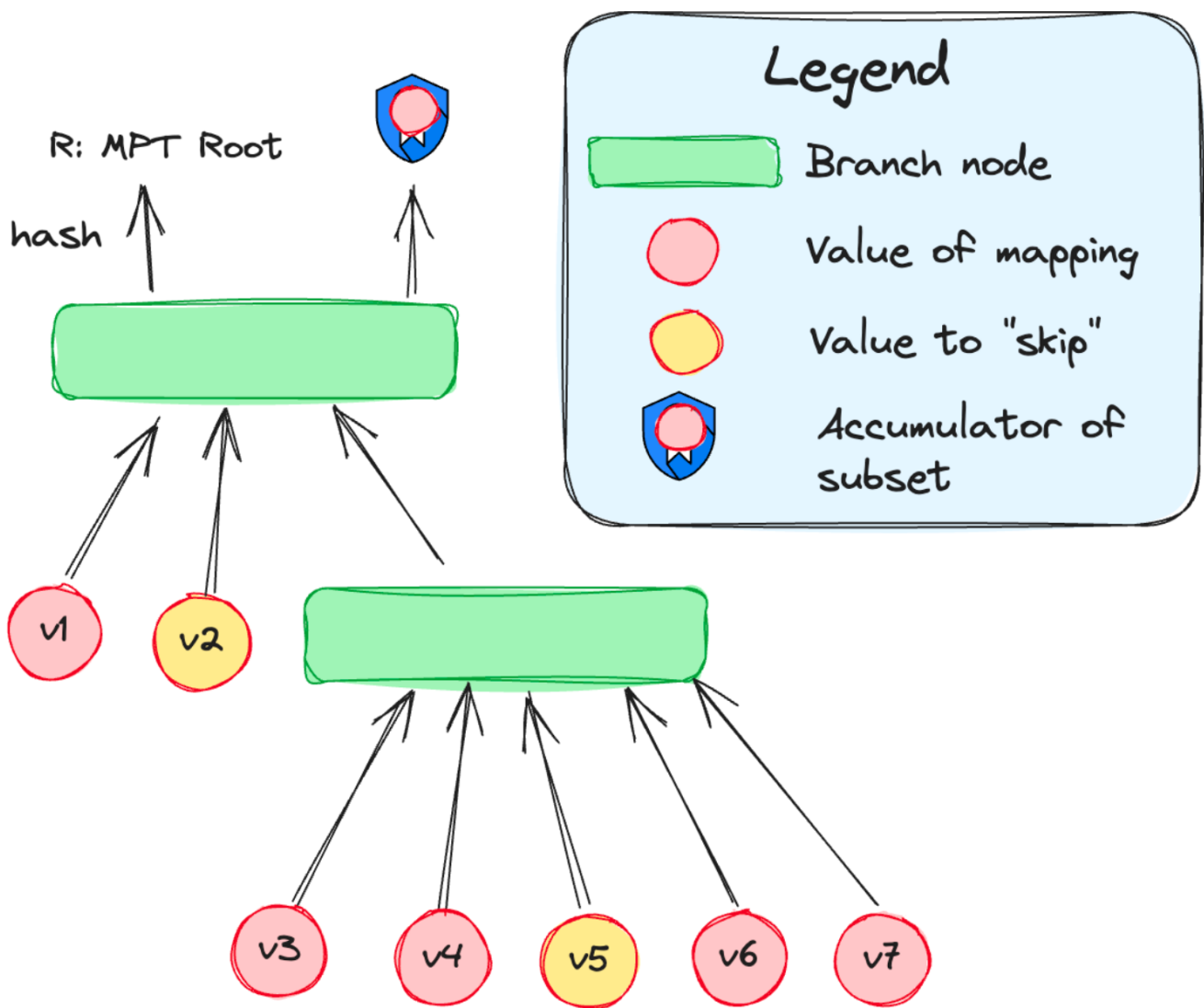
보는 바와 같이, 이 테이블은 동일한 블록에 대해 여러 행을 포함하며, 매핑의 각 항목마다 한 행씩 존재한다. 다음 하위 섹션에서는 스토리지, 상태, 블록 변환의 기술적 세부 내용을 다룬다.

스토리지 데이터베이스

스토리지 데이터베이스는 원래 컨트랙트의 스토리지 트라이 데이터의 부분집합을 포함하는 증명 가능한 등가 데이터 구조이다. 작은 머클 파트리샤 트라이(MPT)와 ZK Coprocessor가 생성하고 유지하는 등가 스토리지 데이터베이스의 구체적 예를 살펴보자.

스토리지 머클 파트리샤 트라이

본질적으로 MPT는 다음 다이어그램과 같다. 여기서는 전체 스토리지 트라이가 아닌, 우리가 증명해야 하는 부분집합만을 대상으로 한다. 예를 들어, 아래 다이어그램은 처리 중인 솔리디티 매핑의 항목들을 빨간색으로 표시한다.



재귀적 증명

이 MPT 트라이의 각 노드는 SNARK/STARK에 의해 “증명”되며, 자식 노드의 증명 유효성과 자식이 올바르게 포함되었음을 검증한다.

즉, 우리는 재귀적 증명을 사용하여 “트리를 구축”하고 있으며, 마지막 증명이 “루트” 증명이다.

부분집합 누적기

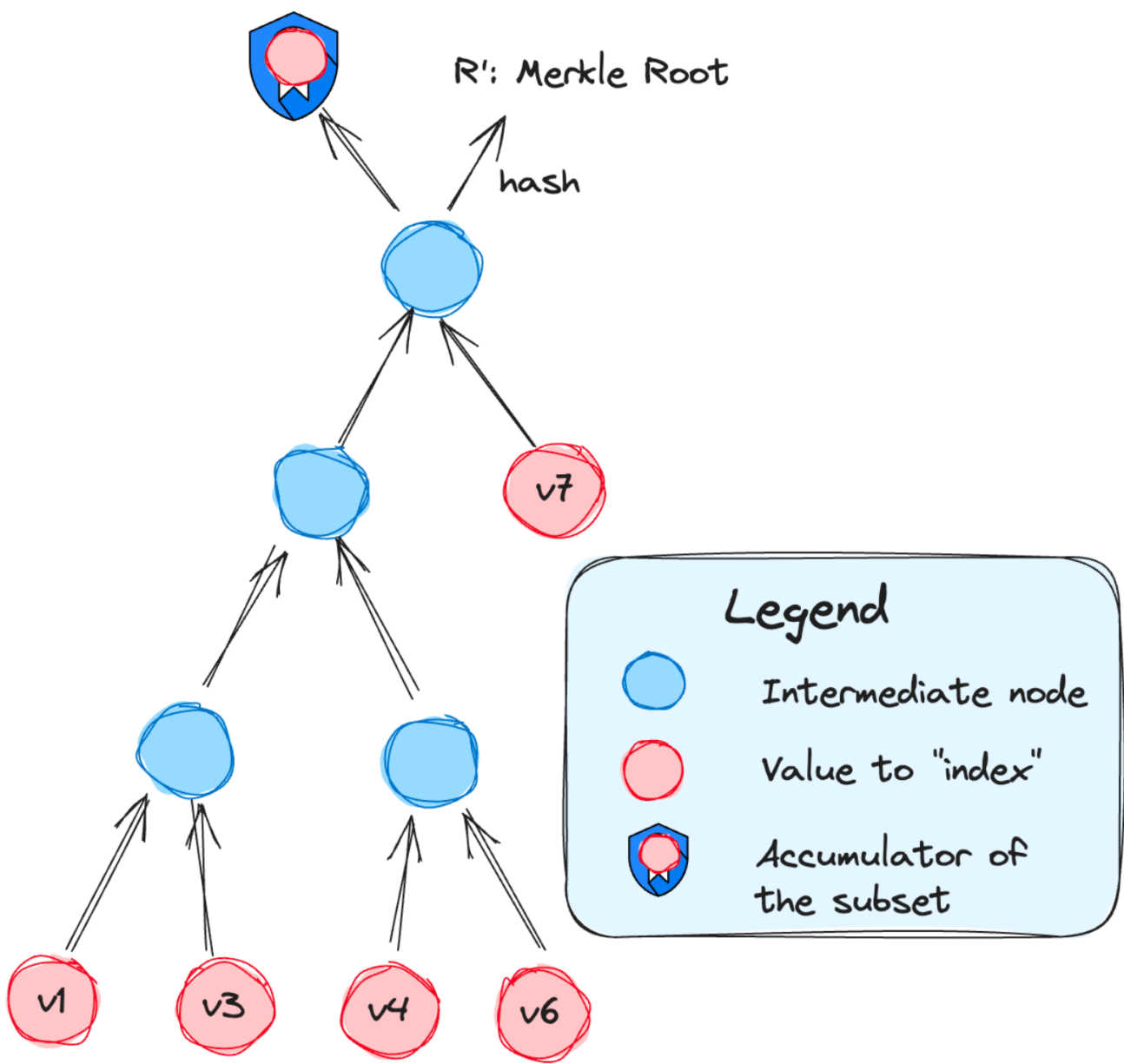
암호학적 증명은 관심 있는 리프의 부분집합에 대한 누적기를 생성하여, 등가 스토리지 데이터베이스에서도 동일한 리프가 양쪽 트리에 포함되어 있음을 증명할 수 있게 한다. 누적기는 값 집합을 간결하게 표현할 수 있다. 해시 함수는 가장 기본적인 누적기 형태이다. 그러나 순서 무관 속성과 SNARK 내 효율적 검증이 필요하기 때문에, 우리는 ecgfp5 곡선 상의 다중 집합 해싱(multiset hashing)을 사용한다.

라그랑주 스토리지 데이터베이스

Coprocessor는 스토리지 트라이에서 관심 있는 키/값 쌍을 추출하고, 이 값들로부터 “ZK 친화적” 머클 트리를 새로 생성하며, 트리 구축 시 동시에 해당 쌍의 다이제스트를 계산한다.

부분집합 누적기

동일한 부분집합을 사용하기 때문에, 누적기의 최종 값은 스토리지 MPT와 ZK Coprocessor 데이터베이스 간에 동일하다. 두 구조 모두 동일한 리프를 “표현”하지만, 누적기는 MPT 트리와의 등가성을 증명하기 위한 특수한 구조이다. 스토리지 데이터베이스는 이후 ZK Coprocessor의 모든 질의에서 사용된다.



ZK Coprocessor 스토리지 데이터베이스 구조

- 머클 트리: ZK Coprocessor 스토리지 데이터베이스는 데이터가 내부 노드가 아닌 리프에 존재하는 머클 트리이므로, MPT처럼 리프 경로에 저장된 부분 키나 RLP 인코딩을 고려할 필요가 없다. 이러한 단순화 덕분에 증명 구조가 더 쉽게 된다.
- 차수(Arity): 우리는 부분집합만을 다루므로 거의 균형 잡힌 완전 이진 트리를 직접 구성할 수 있다. 우리는 차수 = 2를 선택했는데, 이는 항상 두 자식을 강제할 수 있어 비용과 복잡성을 낮춘다. 단, 이는 ZK Coprocessor 데이터베이스의 조정 가능한 매개변수이다.

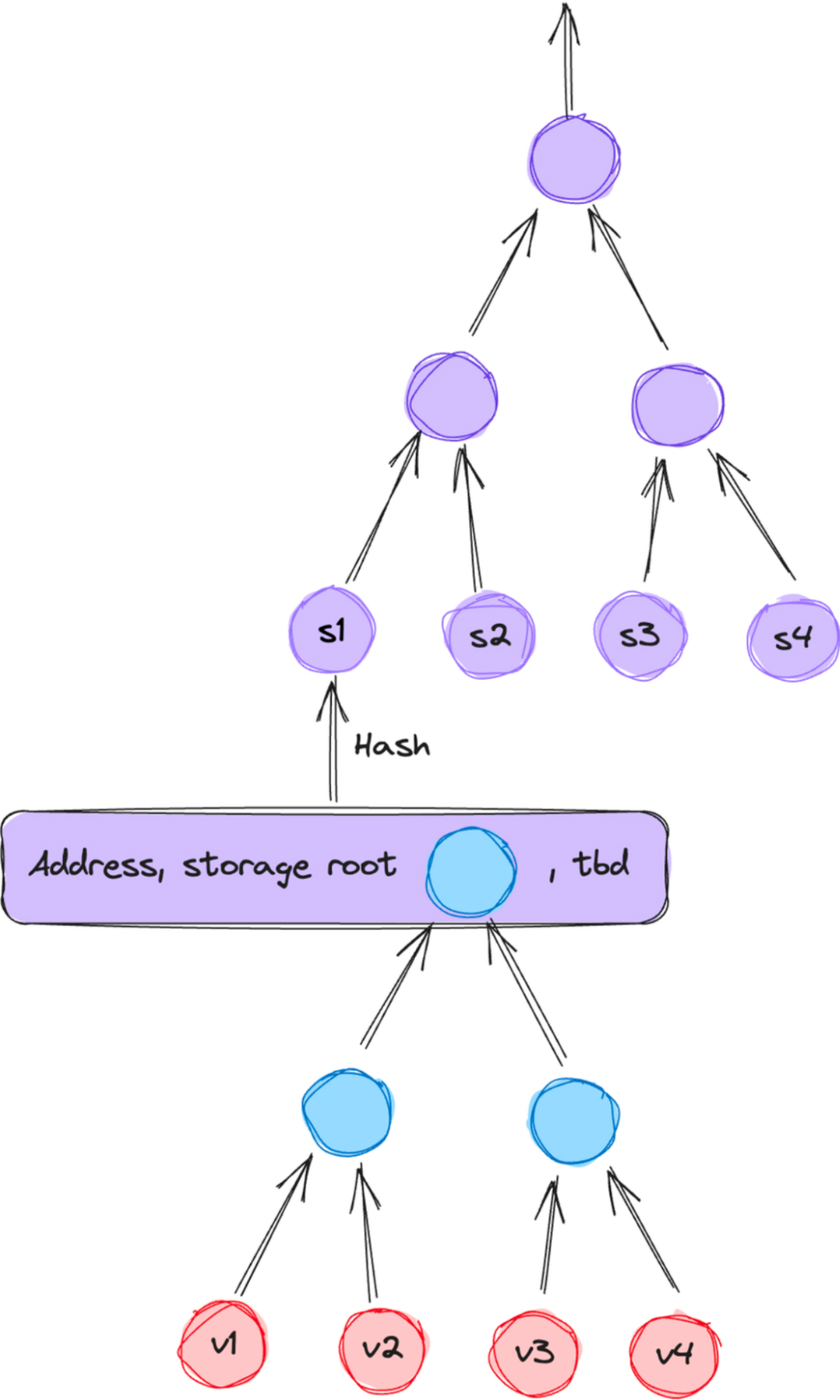
상태 데이터베이스

상태 데이터베이스의 목적은 동일한 구조를 가진 여러 컨트랙트의 스토리지 데이터베이스를 통합하는 것이다. 이는 이더리움의 상태 트라이가 작동하는 방식과 유사하다.

리프 구조

각 리프는 ZK Coprocessor가 인덱싱하는 스마트 컨트랙트에 대한 정보를 유지한다. 이 경우, 컨트랙트의 주소(EVM 주소)와 이전 섹션에서 계산된 해당 스토리지 데이터베이스의 루트만 필요하다.

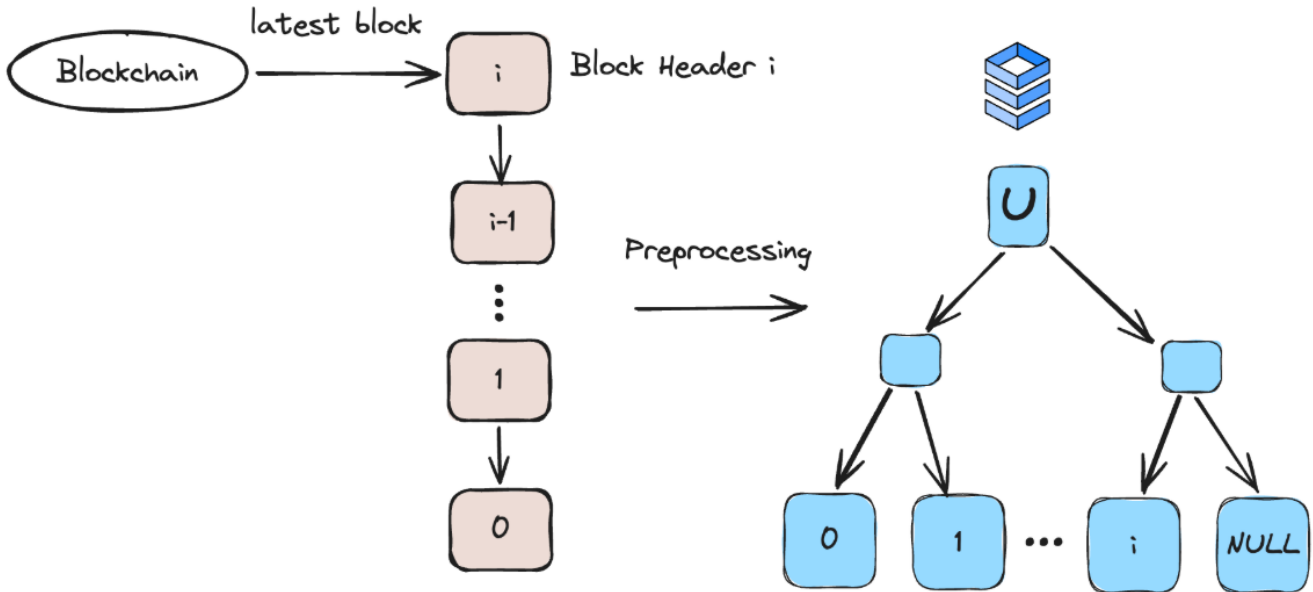
Leaf Structure



블록 데이터베이스

ZK Coprocessor는 이력 질의를 처리하므로, 각 블록에 대한 개별 상태 및 스토리지 트리를 유지해야 한다. 즉, ZK Coprocessor는 각 블록마다 데이터베이스의 스냅샷을 생성한다. 하지만 올바른 변환을 증명하기 위해서는 최신 상태가 대상 블록체인의 해당 블록에 실제로 속하며, 새 블록이 이전 블록에 연속적으로 연결됨을 증명해야 한다.

이를 위해, 매번 새로운 블록이 생성될 때마다 업데이트되는 또 다른 증명 친화적 데이터 구조 내에 연속적인 블록 구조를 재구성한다.



리프 구조

이 트리의 리프는 SNARK 친화적 블록 헤더(SFBH)로 구성된다. 이 헤더에는 다음이 포함된다.

- 블록 번호
- 원본 “블록체인” 블록 해시
- 해당 블록 번호에 대응하는 상태 머클 트리의 루트(이전 섹션에서 계산됨)

블록 트리 생성 및 구조

초기화: 트리는 고정 크기로 설정되며, 예를 들어 $n = 2^{26}$ 개의 리프를 “0”으로 설정한다. 이는 누구나 다시 계산할 수 있는 루트 U_0 를 생성하며, 검증자(스마트 컨트랙트)가 이를 보유한다.

업데이트: 새로운 블록이 생성되고 등가 상태 데이터베이스가 구축된 후, 새로운 SFBH로 블록 데이터베이스를 업데이트할 수 있다. 새 헤더의 트리 내 위치는 블록 번호에 의해 결정된다.

- 예시: 블록 번호 108351부터 처리하기 시작한다면, 블록 번호 108351+10의 헤더는 블록 데이터베이스의 10번째 인덱스에 설정된다.
- ⚠️ 암호학적 증명: 트리를 업데이트할 때마다 재귀적 증명 p 가 생성된다. 이 증명은 원래 루트 U_0 에서 현재 블록 번호 U_i 로 트리가 올바르게 업데이트되었음을 보여준다. 이 증명은 “등가 데이터베이스가 원본 블록체인과 동일한 데이터를 포함함”을 보여주는 최종 증명이다.

ZK MapReduce

라그랑주 네트워크의 핵심은 MapReduce 방식의 분산 연산을 수행하는 것이다. 다음 하위 섹션에서는 이를 자세히 다룬다.

MapReduce 기초

라그랑주의 증명 네트워크는 분산 MapReduce 스택처럼 구축되어 있으며, 그 위에 계산의 검증 가능성을 추가한다. 대규모 데이터베이스에 대한 연산을 수행할 때 여러 아키텍처 선택지가 있다. 단일 서버가 모든 데이터를 다운로드하여 전체 계산을 로컬에서 수행할 수도 있지만, 이는 매우 강력한 머신을 필요로 하며 대역폭 측면에서 비용이 크다. 라그랑주 네트워크는 유명한 MapReduce 프레임워크의 철학을 따르며, 데이터가 존재하는 위치로 계산을 가져오는 방식을 취한다. MapReduce는 대규모 데이터베이스의 작은 청크 단위로 계산을 수행하고, 각기 다른 머신에서 실행된 후 여러 단계의 집계를 통해 결과를 결합한다. 전반적으로 이 계산은 두 단계로 나뉜다.

1. 각 머신은 자신의 데이터 청크에 대해 map 연산을 수행하여 키-값 쌍 집합으로 변환한다.
2. shuffle 결과는 reduce 연산으로 전달되어, 여러 map 연산의 출력을 하나의 결과로 집계한다. 이 연산은 많은 청크에 대해 병렬로 수행되며 단일 결과가 생성될 때까지 반복된다.

MapReduce의 주요 장점은 확장성이다. 계산이 여러 머신에 분산되므로, 단일 머신에서는 처리할 수 없는 대규모 데이터셋을 처리할 수 있다. 또한, MapReduce의 분산 특성은 계산을 시간적으로 확장하는 대신 수평적으로 확장할 수 있게 한다.

ZK MapReduce (ZKMR)

지난 1년 동안 =Nil; Foundation, RiscZero, Polygon Miden과 함께 대중적인 ZKVM이 급증했지만, 이들 각각은 범용 순차 실행 모델에 집중해 왔다. 이에 반해, 라그랑주의 ZK MapReduce(ZKMR)는 재귀적 증명을 활용하여 대량의 온체인 상태 데이터에 대한 분산 계산의 정당성을 증명하는 MapReduce의 확장판이다.

이는 분산 계산 작업의 map 또는 reduce 단계 중 각 증명자에 대해 계산의 정확성에 대한 증명을 생성함으로써 달성된다. 이러한 증명은 재귀적으로 합성되어 전체 분산 계산 워크플로의 유효성에 대한 단일 증명을 구성할 수 있다. 다시 말해, 더 작은 계산들의 증명을 결합하여 전체 계산에 대한 증명을 만들 수 있다. 이는 여러 단계 또는 하위 계산을 요구하는 대규모 데이터셋에 대한 복잡한 계산에서 ZKMR이 효율적으로 확장되도록 한다.

ZKMR의 효율성 개선

ZKMR 접근법은 전통적인 순차 계산보다 더 복잡해 보일 수 있지만, 그 성능은 시간에 대해 수직적으로가 아니라 병렬 머신의 수에 대해 수평적으로 확장된다.

오프체인, 앱 롤업 L3, 대체 L1 등 확장성 솔루션 전반에 걸쳐 상태 저장소를 계속 분할함에 따라, 생성되는 온체인 데이터의 양은 기하급수적으로 증가하고 분절되고 있다. 아주 가까운 미래에 질문은 이렇게 바뀔 수 있다. 100개의 서로 다른 앱 롤업에 배포된 한 DEX에서 1주 TWAP을 계산하기 위해 어느 정도의 데이터를 처리해야 하는가?

요약하면, 순차 계산은 범용 애플리케이션 개발에는 매우 효율적이고 표현력이 풍부하지만, 대규모 데이터셋의 분석과 처리에는 최적화되어 있지 않다. 분산 컴퓨팅의 효율성은 Web2에서 지배적이었던 빅데이터 처리의 표준이 되었다. 영지식 맥락에서는, 확장성과 결합 허용성이 신뢰 불요의 빅데이터 애플리케이션 —예를 들어 복잡한 온체인 가격, 변동성, 유동성 분석—을 처리하는 이상적인 백본이 된다.

Lagrange ZK Coprocessor의 예정 기능

이 섹션은 오프체인 데이터, 배치 질의, 리시트 트라이, 서브쿼리, 결제에 초점을 맞춘 ZK Coprocessor의 예정된 기능에 대한 간단한 소개를 제공한다. 곧 지원될 예정인 이러한 개선 사항은 기존 시스템의 역량을 확장하는 것을 목표로 한다.

- 오프체인 데이터 - 사용자 주도 데이터 통합: 오프체인 데이터 도입은 사용자가 API를 통해 CSV 형식의 자체 데이터를 업로드할 수 있게 함으로써 의미 있는 확장을 제공한다. 이 기능은 검증 가능한 데이터베이스에 인덱싱·저장되는 개인 오프체인 테이블의 생성을 가능하게 하며, 이후 영지식 증명을 사용해 온체인에서 질의하고 증명할 수 있다. 이는 검증 가능한 질의를 위한 개인 데이터 통합을 가능하게 하여, 현재 기능 세트에 주목할 만한 유연성을 추가한다.
- 배치 질의 - 병렬 처리를 통한 효율성: 배치 질의는 맞춤형 단일 회로에서 검증 가능한 데이터베이스의 여러 행을 한 번에 처리함으로써 질의 시간을 최적화하고 계산 오버헤드를 줄이도록 설계되었다. 이 접근법은 개별 질의 처리 비용의 병목을 해결하여 ZK Coprocessor의 확장성에 특히 유익하다.
- 고급 질의 - 조인 및 서브쿼리 처리: ZK Coprocessor는 온체인 및 오프체인 데이터 모두에 대해 조인과 서브쿼리를 포함한 고급 질의를 지원하도록 확장될 것이다. 이 향상은 정교한 질의가 필요한 사용자에게 중요하며, 효율적인 증명 생성을 위한 지속적인 연구로 뒷받침된다.
- 리시트 트라이 - 이벤트 로그로 질의 범위 확장: 리시트 트라이 지원은 트랜잭션 리시트의 로그에 저장된 솔리디티 스마트 컨트랙트가 발생시킨 이벤트를 인덱싱하고 질의할 수 있는 능력을 도입한다. 이 기능은 정적 스토리지를 넘어 컨트랙트 상호작용에 대한 통찰을 제공하여, 검증 가능한 질의에 사용할 수 있는 데이터를 크게 확장한다.
- 결제 - 라그랑주 생태계에서 운영자와 사용자 간의 가치 이전: 결제는 EigenLayer 운영자, ZK Coprocessor 질의 가능한 테이블 스폰서, 그리고 해당 테이블에 대해 질의를 수행하는 고객을 위해 활성화될 것이다. 이러한 결제 통합은 확장성과 신뢰를 지원하는 견고한 금융 생태계를 보장한다.

다가오는 ZK Coprocessor 버전은 오프체인 데이터 통합, 효율적인 질의 배치, 리시트 트라이를 통한 이벤트 질의 확장, 고급 질의 지원, 견고한 결제 시스템을 가능하게 하는 중대한 진전을 약속한다. 이러한 개선 사항은 데이터 기반 탈중앙화 애플리케이션을 위한 강력한 도구를 제공할 것이다.