Nebulas 기술백서

가치 기반 블록체인 운영 및 검색엔진

Nebulas 재단

2018년 4월

v1.0.2

초록(Abstract)

비트코인은 최초로 탈 중앙화된 “개인간 디지털 화폐 시스템”을 도입하여 우리로 하여금 블록체인이라는 새로운 영역에 대해 눈뜨게 하였으며, 이더리움은 “스마트 컨트랙트”를 선보임으로써 블록체인 기술의 개발을 위한 보다 나은 환경을 구축하는데 크게 기여를 했다. 블록체인 산업은 지난 3년간 급속도로 성장했으며, 분산 어플리케이션과 이에 대한 수요가 빠르게 증가하고 있다.

그러나, 분산 어플리케이션, 스마트 컨트랙트, 시스템 시나리오 등이 나날이 늘어나고 있는 가운데 사용자가 블록체인에 대한 구체적이고 본인에게 알 맞은 데이터를 찾는 것은 점점 더 어려워지고 있다. 또한, 블록체인상의 취약점이 발견되어도 스마트 컨트랙트의 영구성으로 인해 단순한 업데이트가 아닌 블록체인 자체를 여러 형태로 나누어 업데이트할 수 밖에 없는 행위는 이미 넘치는 데이터로 인해 혼란을 겪을 사용자 경험을 더욱 악화시키기만 한다. 따라서, 현 블록체인 기술은 1)블록체인 내 데이터의 가치 측정 매커니즘의 확립, 2)자체적으로 진화하는 능력 및 최신 기술을 블록체인에 통합 할 수 있는 능력, 3) 블록체인 생태계 내의 모든 참여자를 위한 건강하고 장기적인 생태계의 구축이라는 세 가지 주요 도전에 직면했다.

Nebulas는 이러한 도전과제를 해결하고자 한다. 본 백서는 Nebulas 블록체인의 기술적 설계 이데올로기와 원칙을 설명하고자 한다. Nebulas의 블록체인은 다음을 포함한다:

- **Nebulas Rank(NR)(§2).** 블록체인 내 데이터의 가치를 측정하는 새로운 매커니즘이다. NR는 Nebulas 블록체인 플랫폼에서 사용되는 주소 및 컨트랙트의 유동성, 전파성 및 상호운용성을 고려하여 가치를 측정한다. 공개적이고, 연산이 가능하며, 결정론적인 시스템이다. 새로운 랭킹 시스템을 통해, Nebulas 블록체인에서 실제 사용되는 분산 어플리케이션이 증가하게 될 것이다.

- **Nebulas Force(NF)(§3).** 핵심 프로토콜 및 스마트 컨트랙트를 주요 체인에서 직접 업그레이드 하는 기능을 지원한다. 실질적인 활용을 위한 준비를 마친다면, 이를 통해 Nebulas는 자가적으로 진화하고 최신의 기술을 주요 체인에 통합할 수 있을 것이다. 사용자는 NF를 사용하여 빠른 연산 반복 작업을 통해 풍부한 어플리케이션을 구축 할 수 있으며, 이러한 어플리케이션은 시장 변화 또는 커뮤니티 피드백에 역동적으로 대응할 수 있다.

- **Developer Incentive Protocol(DIP)(§4)**. Nebulas블록체인 생태계를 보다 나은 방식으로 구축하기 위해 고안되었다. NR를 기반으로 측정된 가치의 순위에 따라 사용자는 NAS(Nebulas 토큰)을 인센티브로 제공받게 될 것이다. DIP를 통해 높은 순위에 위치한 어플리케이션에 대한 인센티브를 제공하고 개발자가 Nebulas 생태계 안에서 지속적으로 가치를 창출할 수 있도록 장려할 것이다.

- **Proof of Devotion(PoD) 합의 알고리즘(§5)**. 건강한 생태계를 구축하기 위해, Nebulas만의 새로운 합의 알고리즘으로 신속성, 비가역성, 공정성이라는 세 가지 주요 포인트를 제안한다.

- 분산 어플리케이션을 위한 검색 엔진(§6). Nebulas는 랭킹 시스템을 기반으로 분산 어플리케이션을 위한 새로운 유형의 검색 엔진을 구성한다. 사용자는 네뷸러스 엔진을 사용하여 적시에 유용한 분산 어플리케이션과 블록체인에 대한 데이터를 검색할 수 있다.

목차

용어해설

BFT Byzantine Fault Tolerant)

DIP Developer Incentive Protocol

NF Nebulas Force

NNS Nebulas Name Service

NR Nebulas Rank

NVM NebulasVirtual Machine

PoD Proof of Devotion

PoI Proof of Importance

PoS Proof of Stake

PoW Proof of Work

SCR Smart Contract Rank

SCS Smart Contract Score

WAA Weekly Active Addresses

1 서론

1.1 블록체인 기술 소개

블록체인 기술은 사토시 나카모토(Satoshi Nakamoto)가 2008년에 개념화한 최초의 분산 디지털 화폐인 비트코인에서 파생되었다 [37]. 비트코인은 특정 기관에서 발행되지 않고, 분산 원장 시스템의 일관성을 보장하기 위해 특정 알고리즘과 대규모 컴퓨팅을 통해 생성된다. 이더리움 [10]은 더 나아가 퍼블릭 블록체인 기반 컴퓨팅 플랫폼에 튜링 완전 언어를 제공한다. 데이터 암호화, 타임 스탬프, 분산 합의, 경제적 인센티브의 구성 요소를 통해 블록체인 기술은 Peer-to-Peer 트랜잭션, 조정, 공동 작업을 노드가 서로 신뢰할 필요 없는 분산 시스템에서 실제로 구현하여 고비용, 저효율, 불안정한 데이터 저장과 같은 중앙 집중식 기관이 가진 공동의 문제를 해결한다.

블록체인 기술 자체가 새로운 기술 혁신이 아니며, Peer-to-Peer 통신, 암호화, 블록체인 데이터 구조 등 일련의 기술을 결합한 기술적 혁신이라는 점에 주의해야 한다.

1.2 사업 및 기술적 난제

보다 많은 이들이 분산 자치 시스템 개발에 참여함에 따라 전 세계에서 암호화 디지털 자산의 가치가 900억 달러에 이르게 되었으며 블록체인 프로젝트는 2,000개 이상으로 급격히 증가했다. 블록체인 사용자 및 디지털 자산 소유자 수는 2013년 초 200만에서 2017년 초 2,000만으로 급격히 증가했다. 2020년까지 블록체인 사용자 및 디지털 자산 소유자의 수는 2억 명을 상회할 것으로 예상되며 2025년에는 10억 명에 이를 것으로 예상된다.

블록체인 기술에 대한 뜨거운 관심으로 인해 블록체인 기반 어플리케이션과 사용 사례가 증가하고 있다. 이러한 사용 사례는 디지털 통화에서부터 이더리움(Ethereum)과 리플(Ripple)이 개발한 글로벌 Settlement Layer은 개발한 스마트 컨트랙트 등과 같이 점차적으로 더 광범위하게 확장되었다. 이렇듯 다양한 응용 프로그램 사용 사례로 인해 기본적인 블록체인 자체에 대한 요구와 과제가 증가하고 있다.

가치 측정. 기존 블록체인의 핵심 과제 중 하나는 가치 측정 방법이 부족하다는 점이다. 블록체인 생태계는 사용자와 스마트 컨트랙트 모두의 가치를 측정하기 위해 보편적 가치 측정이 필요하다. 상위 계층 어플리케이션은 이러한 보편적 가치 척도를 토대로 특정 활용 사례에서 보다 깊은 가치를 추구할 수 있다. 이러한 점에서 미래 비즈니스 모델의 혁신은 풍부해질 것이다. 이러한 혁신은 인터넷의 세계에서 구글(Google)의 부상을 연상케 한다.

블록체인 시스템 업그레이드. 일반적인 소프트웨어와 달리 분산 블록체인 시스템은 사용자가 클라이언트와 프로토콜을 업그레이드하도록 강요할 수 없다. 따라서 블록체인 시스템의 프로토콜 업그레이드는 종종 "하드 포크(hard fork)” 또는 "소프트 포크(soft fork)”를 통해 커뮤니티에 막대한 손실을 가져오고 이로 인해 시스템의 응용성은 더욱 제한된다. 비트코인의 경우, 커뮤니티 내의 블록 스케일링 논쟁에 대한 논란이 여전히 무성하여 비트코인 프로토콜의 발전을 방해하고 있다. 비트코인 블록체인의 심각한 용량 부족으로 트랜잭션 풀이 블록에 기록될 때까지 약 백만 건의 트랜잭션이 대기중인 상황이 발생했다. 사용자는 종종 사용자 경험에 심각한 영향을 미치는 " 트랜잭션 가속화 수수료"를 추가로 지불해야 한다. 게다가 이더리움의 “하드 포크”가 DAO 문제에 일시적인 해결책을 제공하긴 했지만, ETH 와 ETC “자산 중복”이나 커뮤니티 분열과 같은 의도치 않은 “부작용”도 발생시켰다.

블록체인 어플리케이션 생태계 구축. 블록체인 상에서 분산 어플리케이션(DApps)이 급속히 증가하는 가운데, 건전한 생태계가 보다 나은 사용자 경험의 열쇠가 되었다. 항상 고려해야할 점은 블록체인 어플리케이션의 대량 수집으로부터 사용자가 원하는 DApp을 찾도록 돕는 방법, 보다 많은 개발자가 사용자를 위해 더 많은 DApp을 개발하도록 장려하는 방법, 개발자들이 DApp을 구축할 수 있도록 최상의 환경으을 제공하는 방법이다. 이더리움의 예를 들어보자. 이미 이더리움 상에는 수십만 개의 어플리케이션이 구축되었다. 하지만 이러한 어플리케이션이 애플 앱스토어의 App만큼 증가하게 된다면 DApp을 검색하고 찾는 일은 어려운 과제가 될 것이다.

1.3 Nebulas 설계 원칙

Nebulas는 인센티브 기반의 스스로 진화하는 블록체인을 설계하여 이러한 과제를 해결하고자 한다. 설계 원칙은 다음과 같다:

- **가치 측정을 정의하기 위한 공정한 랭킹 알고리즘**

보다 높은 차원의 정보 식별을 가능게 하려면, 블록체인 영역에서 블록체인의 최하위 계층에 위치한 데이터의 가치를 측정하기 위한 보편적 가치 측정 매커니즘이 필요하다. Nebulas는 Nebulas Rank(NR)(§2) 알고리즘(구글의 페이지랭크 [9][45]와 유사)을 제안한다. NR 알고리즘은 유동성과 전파성(속도, 폭 및 깊이)을 블록체인에 결합하여 블록체인 사용자에게 공정한 순위를 제공한다. NR은 블록체인 생태계에서의 가치 척도이며, 개발자는 NR을 통해 다양한 시나리오에서 각 사용자, 스마트 컨트랙트, DApp의 중요성을 측정 할 수 있다. NR은 엄청난 상업적 잠재력이 있으며, 검색, 추천, 광고 및 기타 분야에서도 사용될 수 있다.

- **스스로 진화하는 블록체인 시스템 및 응용프로그램**

탄탄한 시스템과 그 시스템 상의 어플리케이션은 특별한 개입 없이 스스로 진화할 수 있고, 계산이 빠르며, 뛰어난 탄력성 및 향상된 사용자 경험을 제공할 수 있어야 한다. Nebulas는 스스로 진화하는 능력을 Nebulas Force(NF)(§3참조)라고 칭한다. Nebulas 시스템 아키텍처에서는 기본 프로토콜이 블록체인 상의 데이터에 포함되며, 데이터 추가를 통해 자유로이 업그레이드가 가능하게끔 설계했다.

Nebulas 어플리케이션(스마트 컨트랙트)의 경우, Nebulas는 스마트 컨트랙트 최하위 계층 스토리지에서 상태 변수에 대한 크로스 컨트랙트 액세스를 가능하게 함으로써 스마트 컨트랙트 업그레이드를 가능하게 했다. 스스로 진화하는 Nebulas 블록체인은 개발 및 생존 가능성 측면에서 다른 퍼블릭 블록체인보다 유리하다. 또한 개발자는 업그레이드를 통해 취약점과 보완점에 대해 보다 신속하게 대응하고 해킹으로 인한 막대한 손실을 방지할 수 있다

- **블록체인 응용프로그램 생태계 구축**

Nebulas는 블록체인 상의 계정 기여도를 기반으로 하는 PoD(§5 참조) 합의 알고리즘을 개발했다. PoD 알고리즘은 가치 측정 수단으로 NR을 사용하여 생태계에 대한 기여도가 가장 큰 계정를 식별하고 장부 기록 독점을 막기 위해 북키퍼가 될 수 있는 기회를 평등하게 부여한다. 또한 PoS에 경제적 패널티를 통합하여 퍼블릭 블록체인에 나타나는 악의적인 피해를 막아 생태계의 자유를 가능하게 한다. PoD의 주요 특징은 PoS 및 PoI보다 합의 속도가 더 빠르고 강력한 반(反) 부정 기능을 내재하고 있다는 점이다.

또한 Nebulas는 스마트 컨트랙트와 DApp 개발자를 위한 Developer Incentive Protocol(DIP, 개발자 인센티브 프로토콜) (§4 참조)를 개발하고 있다. DIP는 스마트 컨트랙트와 DApp 개발자 커뮤니티에 대한 기여도에 따라 인센티브를 보상하는 시스템을 목표로 한다. NR 메커니즘을 기반으로 Nebulas는 사용자가 블록체인에서 보다 높은 가치의 어플리케이션을 더 쉽게 탐색 할 수 있도록 추가적으로 검색 엔진 (§6 참조)을 포함한다.

이더리움은 대규모 생태계를 갖춘 성공적인 퍼블릭 블록체인 플랫폼이다. 따라서 Nebulas는 이더리움의 탁월한 디자인과 설계를 학습하고 스마트 컨트랙트를 완벽하게 지원하여 이더리움 기반 어플리케이션이 이주 비용 없이 Nebulas에서 실행될 수 있기를 바란다.

위 설계 원칙을 기반으로 Nebulas는 가치 측정을 기반으로 하는 블록체인 운영 체제와 검색 엔진을 구축하기 위해 노력할 것이다. 본 백서는 Nebulas의 기술에 대해 자세히 서술하고자 한다. §2는 가치 척도 모델인 Nebulas Rank와 그와 관련한 랭킹 알고리즘을 설명한다. §3은 Nebulas의 자체 진화 능력인 Nebulas Force를 설명한다. §4, §5, §6, §7은 블록체인 어플리케이션 생태계에 대한 Nebulas의 컨셉과 설계를 다룬다. §8에서는 Nebulas의 토큰인 NAS를 다룬다.

2 Nebulas Rank

2.1 Nebulas Rank 개요

현재 블록체인 기술과 커뮤니티는 대규모의 생태계로 성장했다. 그러나, 여전히 블록체인 영역에 대한 많은 이들의 인식은 비교적 낮은 편이다. 블록체인에서 주체(예: 사용자 주소)의 가치를 평가할 수 있는 합리적인 방법이 아직 없다. 따라서 Nebulas는 모든 블록체인 데이터에 대한 보편적 가치 측정을 마련하고자 한다. 블록체인 상의 활동을 탐색하고 활용함으로써 각 주체(사용자 주소)의 가치를 정량화할 수 있는 Nebulas Rank를 구현하고자 한다. Nebulas Rank는 다음 목적으로 설계되었다:

- 합의 알고리즘 (§5 참조), DIP (§4 참조), 블록체인 검색 엔진 (§6 참조) 등과 같은 많은 기본 시나리오에 대한 핵심 가치 및 핵심 알고리즘의 기본 척도로 사용하고자 한다.
- 사업 결정 및 연구 활동을 보다 효과적으로 안내할 수 있도록 다양한 데이터에 대한 가치 측정 및 블록체인 생태계에 대한 심층적인 통찰력을 고취하고자 한다.

상기 언급된 목표에 따라 **Nebulas Rank**의 가치 측정을 다음 세 가지 차원에서 정의한다:

- **유동성.** 유동성은 트랜잭션의 빈도와 규모는 Nebulas Rank가 측정하는 첫 번째 차원이다. 금융자산 유동성의 경우, 본질적으로 자산 유동성을 통해 사회적인 자원을 최적화하여 경제 발전을 촉진하는 사회적 활동이다. 블록체인은 금융자산이 유동적일 수 있게 네트워크를 구축한다. 트랜잭션 빈도와 트랜잭션 규모가 증가하면 유동성이 향상되고, 유동성이 높아지면 트랜잭션 가치와 규모가 증가하여 긍정적인 피드백의 완전한 메커니즘이 형성된다.

- **전파성.** 유동성의 범위와 깊이를 나타내는 전파성은 Nebulas Rank가 측정하는 두 번째 차원이다. 소셜 네트워크에서의 전파, 속도, 범위, 정보 전달 및 연결의 깊이는 소셜 네트워크의 품질과 사용자 증가의 모니터링에 쓰이는 핵심 지표이다. 블록체인 세계에 같은 패턴이 있다. 블록체인 영역에서도 이러한 유사한 패턴을 볼 수 있다. 블록체인 영역에서의 강력한 전파는 자산 유동성의 범위와 깊이를 나타내며, 이러한 전파성은 블록체인에서 자산의 품질 및 규모를 증대시킬 수 있다 .

- **상호운용성.** 상호운용성은 Nebulas Rank가 측정하는 세 번째 차원이다. 인터넷의 초기 단계에는 단순한 웹사이트와 격리된 정보만 존재했다. 현재는 서로 다른 플랫폼의 데이터를 네트워크를 통해 전달할 수 있으며, 데이터의 고립성은 이미 크게 개선되었다. 이러한 경향은 높은 차원의 관점에서 정보를 인식하는 과정으로 이해할 수 있다. 블록체인 영역 또한 이와 같은 패턴을 따라가겠지만, 그 변화의 속도는 더욱 빠를 것이다. 사용자, 자산, 스마트 컨트랙트와 DApp에 대한 정보는 더욱 풍부해 질 것이며 정보의 상호 운용성은 보다 빈번해질 것이다. 결과적으로 데이터 및 정보의 상호 운용성 매우 중요한 요소라고 할 수 있다.

Nebulas는 블록체인 영역의 "궤적"이 현실 세계보다 더 분명하고 신뢰할 수 있기 때문에, NR의 소스 데이터로 체인의 트랜잭션 기록을 선택했다. "스마트 컨트랙트"의 전송 및 호출과 같이 트랜잭션 데이터는 모두 체인에 기록된다. 그러나 블록체인 영역의 트랜잭션은 특성상 익명으로 이루어지고, 실제 세계보다 데이터 규모가 방대하기 때문에, 블록체인 트랜잭션 데이터용 랭킹 알고리즘을 설계하는 것이 쉽지 않다. Nebulas Rank를 위한 세 가지 주요 특징은 다음과 같다:

- **진실성.** 주체는 순위를 높이기 위해서 일정의 비용을 지불해야 하며, 이를 통해 랭킹 알고리즘은 신뢰할 수 있고 가치 있는 사용자를 식별할 수 있다는 점을 보장한다. 이처럼 신뢰가 가능한 랭킹 알고리즘은 합의 알고리즘 및 DIP와 같은 시나리오에서 긍정적인 피드백 시스템을 실현하기 위해 진실성이 보장된 기여를 장려한다. 한편, 진실성이 보장된 결과는 모든 사용자의 순위별 신분을 나타내기 때문에 의사 결정자가 이를 참고하여 보다 나은 결정을 할 수 있게 도와준다.

- **가산성.** 기본 필드로서 Nebulas Rank의 결과는 가장 빠르고 직접적으로 제공되어야하므로, 계산상의 복잡성이 낮아야 한다.
- **결정성.** 합의 알고리즘 및 DIP와 같은 시나리오에서 요구되는 대로 Nebulas Rank의 계산 결과는 모든 클라이언트에서 동일해야 한다.

다음으로 Nebulas는 Nebulas Rank의 기본 프레임워크를 설계한다. 첫째, 트랜잭션 기록은 그래프의 형태로 표현된다. 트랜잭션 그래프(주체 그래프)에서 모든 노드는 하나의 주체에 매핑되고, 각 엣지는 두 주체간의 전송을 나타낸다[56]. 트랜잭션 그래프는 사용자간의 금전적 흐름이 자산 흐름으로 연결된다는 사실을 구체화한다. 이는 이전에 정의된 유동성 및 전파성의 개념을 나타내는데 도움이 된다. 한편, 그래프의 형태는 계약간의 상호운용성을 명확하게 나타낼 수 있다. 파생된 트랜잭션 그래프를 사용하여 각 노드의 네트워크 트랜잭션 중심으로 순위를 부여한다. Nebulas Rank 시나리오에서 Leader Rank [14][31]는 보다 합리적인 척도이며 구글의 페이지랭크(PageRank)나 NEM의 NCDAwareRank의 성능을 능가한다[38].

2.2 트랜잭션 그래프

본 하위 섹션에서는 트랜잭션 내역에서 트랜잭션 그래프를 파생시키는 방법을 소개한다.

첫째, 모든 시간 t_0에 대해, 우리는 $[t_0 - T, t_0]$ (T는 상수로서 일반적으로 1개월로 설정) 동안 개별 주소 사이에서 모든 유효한 트랜잭션을 취한다. 여기서 모든 트랜잭션은 4-튜플(s, r, a, t)로 s는 소스 주소, r은 대상 주소, a는 전송량, t는 블록 시간이다. 우리는 $a > 0$ 및 $s \neq r$인 경우에만(트랜잭션이 효과적이라고 정의한다. 따라서 $[t_0 - T, t_0]$ 동안의 모든 효과적인 트랜잭션은 4-튜플의 집합으로 나타낼 수 있다.

$$\Theta(t_0) = \{(s, r, a, t) \mid t_0 - T \leq t \leq \wedge\, a > 0 \wedge s \neq t\} \tag{1}$$

다음으로, $\Theta(t_0)$에 기초하여, 가중치가 부여된 단순 그래프 $G = (V, E, W)$를 구축한다. 여기서 노드 집합, 엣지 집합, 엣지 가중치는 각각 V, E, W로 표시된다. V의 모든 노드는 개별 계정의 주소를 나타내며 E의 각 에지는 두 주소 간의 전송 강도를 나타낸다. 엣지는 관련한 모든 트랜잭션의 상위 K 금액을 집계하는 가중치 w_e로 유도되며 지정된다.

$$w_e = \sum_{i=1}^{K} a_i,\ s.t.\ a_i \in A_e \tag{2}$$

A_e는 $\Theta(t_0)$의 s에서 r까지의 모든 트랜잭션 양으로 구성된 순서화된 집합이다.

$$A_e = \{\, a_i \mid e = (s,r) \wedge (s,r,a_i,t) \in \Theta(t_0) \wedge (a_i \geq a_j, \forall_i \leq j)\} \tag{3}$$

또한 $N = |V|$, $M = |E|$에서 $|.|$는 집합의 기수이다. 단순화를 위해 모든 노드는 1과 N 사이의 정수로 표시된다.

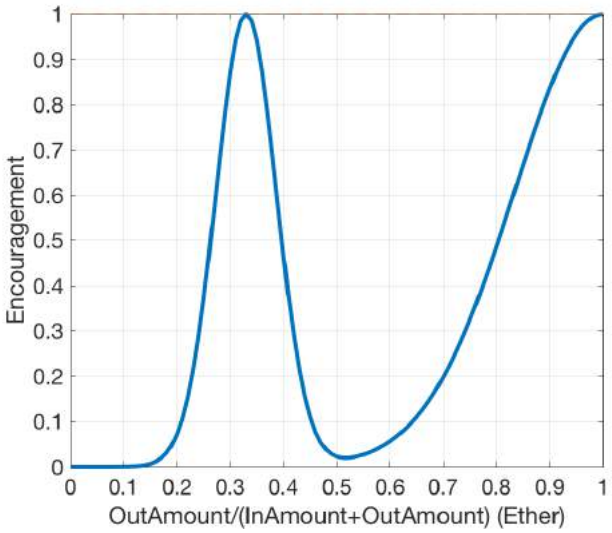


그림. 1: 권장 기능

그런 다음 각 노드에 대해 $[t_0 - T, t_0]$ 구간의 인 트랜스퍼(in-transfers) 및 아웃 트랜스퍼(out-transfers)에 따라 "주화(coinage)"를 계산하고 이를 C_v로 나타낸다. 총 이동량을 바탕으로 그림.1에 나타난 "권장 기능"을 사용하여 권장값을 계산하고 이를 E_v[1]로 표시하며, 대상 노드의 C_v와 E_v를 사용하여 엣지의 중량을 줄인다.

마지막으로 전체 그래프의 가장 약한 연결 요소를 취하고 노드만 해당 구성 요소에 속하도록 한다. 삭제된 노드는 기본적으로 중요도 점수가 가장 낮게 할당된다.

위에서 설명한 그래프 도출은 §2.1에 정의된 "진실성" 속성에 기여하며, 그 증거는 §2.4에 나와 있다.
여기에 설명된 방법을 사용하여, T가 30일이고 $K = 2$인 트랜잭션 그래프가 생성된다. 생성된 그래프는 #3629091(대략 2017년 5월 1일) 블록부터 #3800775(대략 2017년 5월 31일) 블록까지의 이더리움 메인 체인 트랜잭션 데이터를 기반으로 한다. 시각화는 그림.2와 같다. 모든 노드는 정도에 따라 크기가 조정된다. 일부 유명한 교류가 일반적으로 다른 교류보다 많은 계정과 상호 작용한다는 것을 알 수 있다. 게다가 많은 트랜잭션에 기여한 일부 주소의 신원은 여전히 알려지지 않았다.

[1] 권장 기능은 두 개의 정규 분포의 선형 조합으로 표현 될 수 있으며, 이는 전송된 금액이 0일 때 혹은 일부 비율의 금액이 전송되었을 때 최고 값을 출력한다.

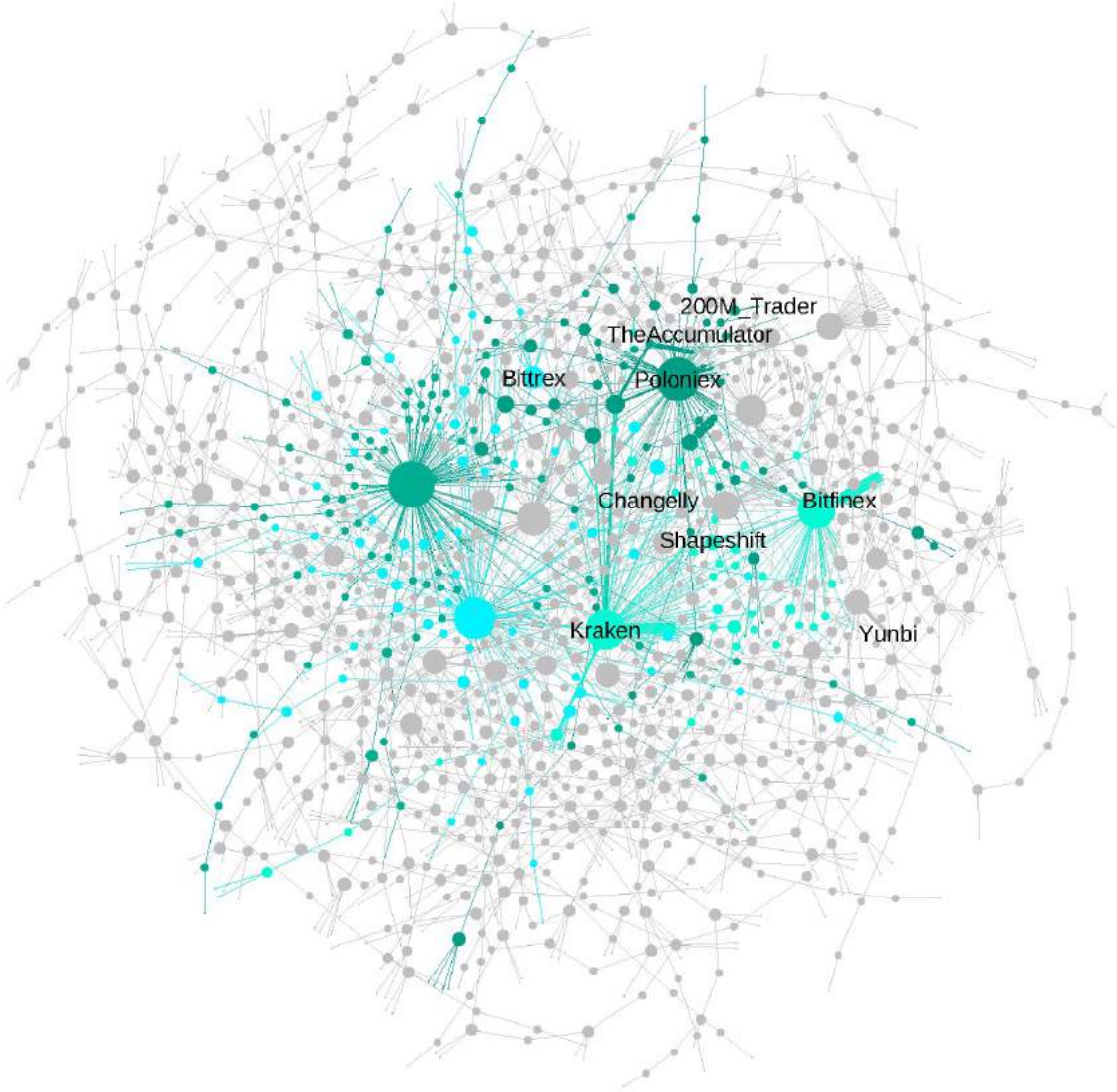


그림. 2: 트랜잭션 그래프(부분) 시각화. 주소에서의 트랜잭션 규모(자본 이동)가 크다는 것은 노드에서 입출력도가 높다는 것을 의미하며 그림에서 지름이 크게 표시된다. 일부 노드는 이더스캔[16]에 따라 태그로 라벨을 붙이게 된다.

2.3 랭킹 알고리즘

하위 섹션에서는 파생 트랜잭션 그래프에서 중요도에 따라 노드 순위를 분배하는 방법을 소개한다. 우리는 리더랭크[14][31]를 주 알고리즘으로 채택한다. 먼저 인덱스 0의 그라운드 노드를 트랜잭션 그래프에 추가한다. 그런 다음 그라운드 노드 0과 다른 모든 노드 i ($1 \leq i \leq N$) 사이에 양방향 링크를 설정하여 다음 공식에 따라 가중치를 부여한다:

$$w_{0,i} = \alpha(max\{\sum_{w_{j,i}\neq 0} w_{j,i} - \sum_{w_{i,j}\neq 0} w_{i,j}, 0\} + \lambda C),\ \forall i \in [1,N] \tag{4}$$

$$w_{0,i} = \beta(\sum_{w_{i,m}\neq 0} w_{i,m} + \mu C), \forall i \in [1,N] \tag{5}$$

C는 집합 $\{w_{i,j} | w_{i,j} \neq 0, 0 \leq i,j \leq N\}$의 중앙값이며, $\alpha, \beta, \mu, \lambda$는 매개변수이다.

가중치 체계는 보다 많은 진입 차수가 있는 노드가 그라운드 노드에서 더 많은 인-링크(in-link)를 수신한다고 설명할 수 있다. 절대 수입이 더 큰 노드, 즉 진출 차수를 제외한 진입 차수가 그라운드 노드에 더 강한 링크를 출력한다.

LeaderRank의 계산 과정은 페이지랭크(PageRank)와 유사하다. 이 점은 마르코프 과정(Markov Process)의 수렴 상태를 계산하는 것으로 이해할 수 있다. 유일한 차이점은 그라운드 노드를 추가 한 후 더 이상 페이지 링크[9][45]의 감쇠 계수를 고려할 필요가 없다는 점이다. 즉, 공식 (7)에 따라 행렬 H를 구성한 후에, 공식(8)에 의해 정의된 초기 설정으로 공식(6)과 같이 수렴될 때까지 계산 과정이 반복된다. 최종적으로 그라운드 노드의 순위 점수는 다른 모든 노드에 균등하게 배분되어 모든 노드의 최종 점수를 산출한다.

$$P^{t+1} = H \times P^t; P^1 = [0, \frac{1}{N}, \frac{1}{N}, \ldots, \frac{1}{N}]^T \tag{6}$$

$$h_{ij} = \frac{w_{(j,i)}}{\sum_k w_{(j,k)}} \tag{7}$$

$$\forall v \in V, P_v^* \leftarrow P_v^* + \frac{P_{\mathcal{G}}^*}{N} \tag{8}$$

LeaderRank가 §2.1에 정의된 값 및 알고리즘 특성 척도를 만족시킬 수 있다고 가정한다.

- 리더랭크의 결과는 네뷸러스 랭크의 "유동성", "전파", "상호 운용성"라는 가치의 척도와 일치하는 자산 교환 네트워크 형식의 동적 균형에서 각 노드의 흐름으로 이해될 수 있다.
- 공식 (5) 와 (4)로 정의된 가중치 체계는 공격을 더 어렵게 만들며, 이는 “진실성" 속성을 만족시킨다. (§2.4의 논의 참조)
- 리더랭크는 누승법(power iteration)으로 계산할 수 있다. 네트워크가 밀도가 매우 희박하기 때문에, 행렬 계산의 복잡성이 낮아야 하고, 이는 "가산성"및 "결정적"이라는 특성을 만족시킨다.

2.4 반(反)부정 기능

진실성, 즉 반부정의 기능은 Nebulas Rank에 있어서 가장 중요하고 어려운 목표이다. 일부 부정행위를 위한 방법은 다음과 같다:

1. 루프 전달(Loop transfer). 공격자는 루프 토폴로지(topology)를 따라 이동하므로 같은 에지를 반복적으로 통과하는 동일한 자금을 허용한다. 이 방법으로 공격자는 관련 엣지의 가중치를 높이고자 한다.
2. 임의의 주소로 이동. 시빌 노드(sybil node)의 진출 차수가 증가하고, 자산의 전파성이 증가된다.

3. 공격자가 제어하는 주소를 사용하여 독립적인 네트워크 구성 요소를 구성하여, 공격자는 중앙 노드로 가장할 수 있다.
4. 신뢰할 수 있는 교환(Exchange) 서비스 주소와 빈번한 상호작용하는 것(즉, 동일한 자금을 신뢰할 수 있는 교환(Exchange) 서비스 주소 안팎으로 반복 전송)을 통해 공격자는 네트워크에서 더 나은 구조적 지위를 얻을 수 있다.

Nebulas Rank는 다음과 같은 매커니즘을 통해 부정행위를 예방한다:

- T 블록의 슬라이딩 윈도우로 인해 공격자는 단기간에 순위를 올릴 수 없다.
- 엣지 가중치는 가장 많은 양의 관련 트랜잭션으로 결정되어, 루프 토폴로지를 따라 전송해도 엣지 가중치를 무제한 증가시킬 수 없다. 한편 §2.2의 샘플 데이터에 따르면 에지의 91%는 각각 2개 미만의 트랜잭션에 해당한다. 따라서 $K = 2$는 루프 전송에 대한 저항인 동시에 엣지에서 강도 정보를 유지하는 합리적인 선택이다.
- 보다 높은 "주화"를 가지려면 사용자가 잠시 동안 자신의 주소에 머물러 있어야 하므로 공격 속도가 느려진다.
- 최대 “권장 가치”를 얻기 위해서는 그림.1에서와 같이 계정은 소득보다는 더 많은 것을 소비하거나 적은 소득 비율을 이동해야 한다. 따라서 자금 흐름 구축 시 공격자의 예금은 급격하게 줄어들게 된다.
- 거대한 구성 요소만 선택되기 때문에 위조 구성 요소를 비롯한 다른 독립 구성 요소가 노이즈로 필터링된다. §2.2의 샘플 데이터에 따르면 453,285개의 노드, 970,577개의 엣지, 1,169 개의 구성 요소가 존재한다. 가장 큰 구성 요소에는 449,746 개의 노드가 있으며 총 노드 수의 99.2%를 차지한다. 두 번째로 큰 구성 요소에는 133 개의 노드만 있으며 0.03%를 차지한다. 따라서 거대한 구성 요소를 취하면, 가능한 네트워크의 정상적인 부분을 유지하면서 노이즈를 필터링할 수 있다.
- 페이지랭크와 NCDawareRank [42]와 같은 웹페이지 랭킹 알고리즘과 비교할 때, 공식(5)와 공식(4)에 의해 정의 된 매커니즘은 소득이 낮은 노드에서 더 “보수적”이다. 즉, 진입차수가 낮은 노드는 그라운드 노드에서 약한 링크를 받는다. 블록체인 트랜잭션 그래프에서는 소득이 낮은 노드가 생성 될 가능성이 높고 다른 임의의 노드로 전환하면 진입 차수가 올라갈 수 없다. 따라서 Nebulas Rank는 부정행위의 어려움을 증가시킬 수 있다.

2017년 5월 이더리움의 트랜잭션 그래프를 바탕으로 다음과 같은 결론을 얻었다.

먼저, Nebulas Rank의 일부 주소는 **표.1**[2]에 나열되어 있다. 트랜잭션 양이 높은 교환 주소나 일부 계정이 상위 노드로 평가된다는 점을 알 수 있다.

[2] 도메인 출처: 이더스캔(Etherscan)[16]

표. 1: Nebulas Rank 상위 10개 주소 및 기타 주소

랭크 (순서)	주소	Nebulas Rank	도메인	출금액(이더/Ether)	입금액 (이더/Ether)
1	0x267be1c1d684f78cb4f6a176c4911b741e4ffdc0	0.449275	Kraken_4	3214232.06	350008.00
2	0xd4c5867cec094721aabc3c4d0fd2f2ac7878c79a	0.093798		58000.00	100947.00
3	0x027beefcbad782faf69fad12dee97ed894c68549	0.049277	QuadrigaCX	207440.11	65606.40
4	0x0ee4e2d09aec35bdf08083b649033ac0a41aa75e	0.046831		56465.00	60087.96
5	0xc257274276a4e539741ca11b590b9447b26a8051	0.037628		1071105.93	1434106.72
6	0xa53e0ca7d246a764993f010d1fde4ad01189f4e6	0.033488		7764.68	3201.00
7	0xf259e51f791e9ed26e89b6cae4a7c6296bfbd0b8	0.033481		3307.00	7731.30
8	0xf195cac8452bcbc836a4d32cfb22235af4ac1e9c	0.026343		10863.87	2315.69
9	0x94435d12c51e19d5b5c8656763f9069d37791a1a	0.024970		12938.58	15858.90
10	0x7580ba923c01783115d79975d6a41b3d38eff8d5	0.021670		263000.00	364793.49
16	0xcafb10ee663f465f9d10588ac44ed20ed608c11e	0.004995	Bitfinex_1	360000.00	1435858.40
51	0xd94c9ff168dc6aebf9b6cc86deff54f3fb0afc33	0.000868	yunbi_1	1179224.74	1202539.53
64	0x70faa28a6b8d6829a4b1e649d26ec9a2a39ba413	0.000590	Shapeshift	52501.81	651933.49

그런 다음 트랜잭션 금액과 Nebulas Rank 사이의 관계를 관찰한다. 보르가티(Borgatti) [5]의 작업에 따르면, 블록체인 전송은 "자금 교환"의 네트워크 흐름으로 이해 될 수 있기 때문에 노드의 차수, 즉 인접 엣지 가중치 합이 이러한 네트워크 흐름에 대한 적절한 중심성 측정기준이다. 각 노드의 관점에서, 차수, 즉 트랜잭션 처리량(전송된 양 더하기 전송 한 양)은 한 홉(hop) 내의 로컬 정보를 나타내며 대응하는 주소에 대한 과거의 자금 흐름을 반영한다. 따라서 랭킹 알고리즘의 기준이 될 수 있다. 트랜잭션 금액과 Nebulas Rank 사이의 관계는 그림 3에서 확인할 수 있다. 낮은 트랜잭션 금액으로 높은 순위를 얻을 수 있는 노드는 없다. 트랜잭션 양이 많은 노드가 높은 순위를 얻으려면 여전히 몇 가지 조건을 충족해야 하며 이는 Nebulas Rank의 진실성을 대략적으로 확인시켜준다.

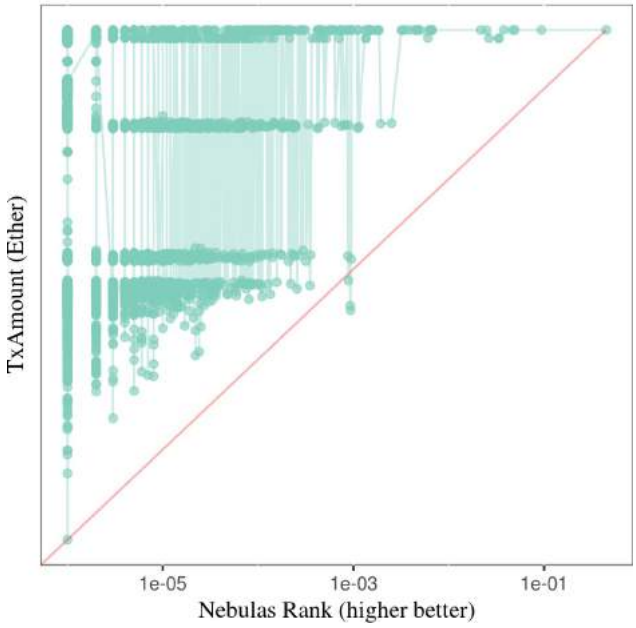


그림.3: Nebulas Rank v.s. 트랜잭션 금액 X축은 순위 값(Rank Value)을 Y축은 트랜잭션 금액을 나타내며 양축 모두 대수 형태이다. 대각선은 트랜잭션 금액과 순위 값이 정비례함을 나타낸다. 좋은 알고리즘은 데이터 포인트가 대각선의 오른쪽 아래에 가능한 한 작게 떨어지도록 하여 트랜잭션 처리율이 낮은 노드가 높은 순위를 가지지 않도록 해야 한다.

위의 간단한 분석에 따르면 처음 세 유형의 조작을 특정 방법으로 효과적으로 필터링할 수 있다고 추론할 수 있다. 따라서 최종적으로 마지막 유형을 시뮬레이션하고 저항 효과를 관찰해야 한다. 공격자는 신뢰할 수 있는 교환(Exchange) 노드 하나를 선택하여 X회 동안 루프 전송을 생성한다. 모든 루프 전송은 2단계로 구성되어 있다. 먼저 공격자가 새로 생성된 임시 주소를 통해 Y 이더(Ether)를 교환(Exchange)으로 전송한다. 그 후, 공격자는 다른 주소를 통해 교환(Exchange)에서 돈을 회수한다. 공격 토폴로지와 프로세스는 그림4에 나와 있다. 이러한 유형의 공격은 교환(Exchange) 서비스가 매우 저렴한 비용으로 모든 노드와 기꺼이 연결하려고 한다는 점을 악용한다.

일반 노드는 교환 노드와 함께 자주 전송할 수도 있지만, 이러한 공격 활동은 유효 화폐 유동성을 향상시키지 못하며 정상적인 활동과 구별되어야 한다.

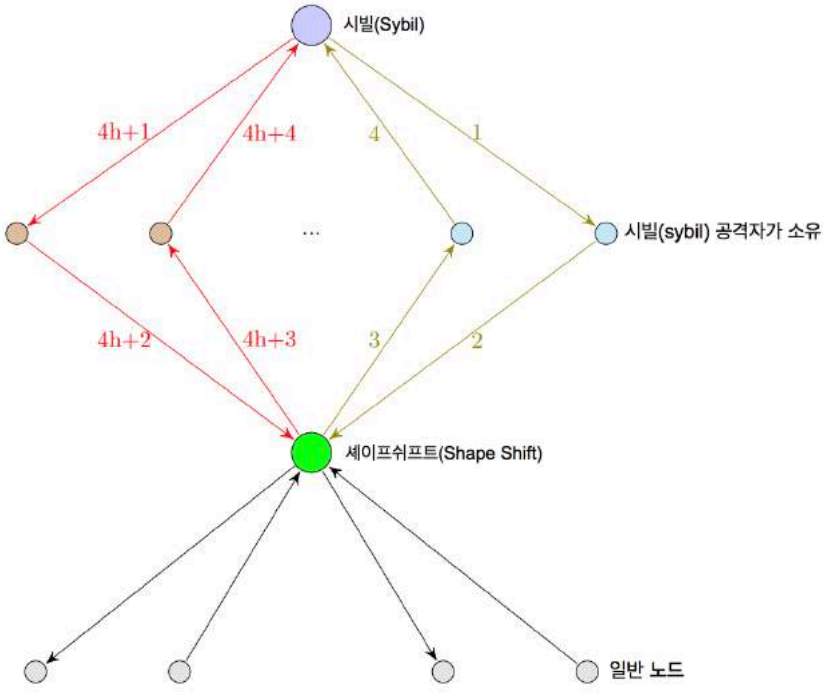
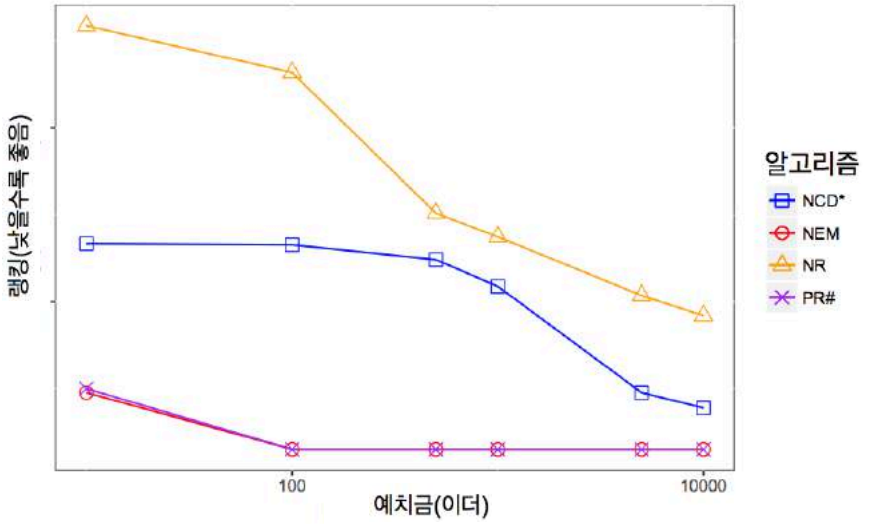
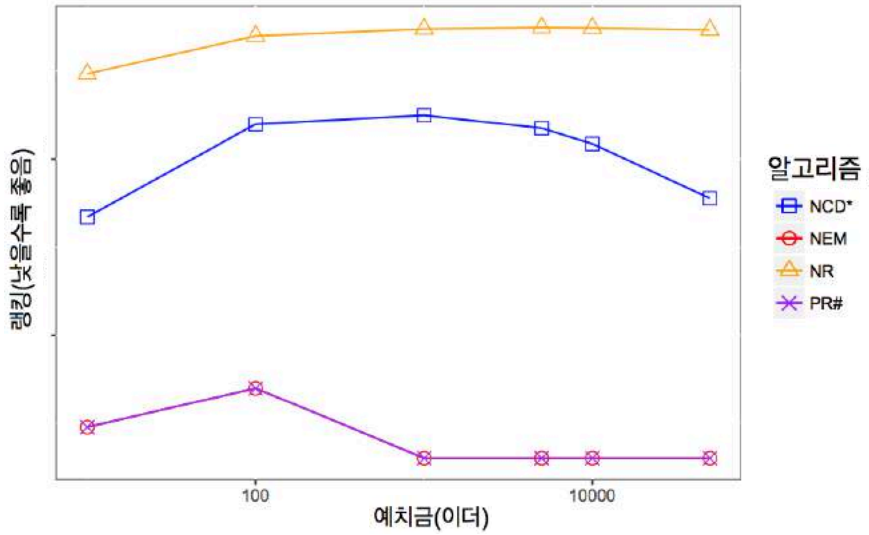


그림.4: 교환 주소를 사용하는 루프 공격의 개략도 첫 번째 및 h번째 루프 공격이 그림에 표시된다. 셰이프시프트가 신뢰할 수 있는 교환 노드로 선택되었다. 엣지 레이블은 시간 순서를 나타낸다. 시빌 공격자와 셰이프시프트 노드가 제어하는 노드 사이의 전송량은 Y 이더(Ether)이다. 조작 중에는 X 루프 전송이 있다.

Nebulas는 셰이프시프트(0x70faa28a6b8d6829a4b1e649d26ec9a2a39ba413)를 신뢰할 수 있는 교환 주소로 선택한다. 결과는 **그림.5**에 나타난다. 1) **그림.5a**에 보여지는 바와 같이, 공격자가 더 많은 자본을 투자하면, 공격자의 순위가 올라가는 것을 막을 수 있는 알고리즘은 없다. 하지만 §2.2에 정의된 트랜잭션 그래프는 공격 효과를 감소시킨다. Nebulas Rank는 트랜잭션 처리량이 높은 노드를 선호할 뿐만 아니라 부정행위를 어느 정도 예방할 수 있다. 2) **그림.5b**에서 볼 수 있듯이, 공격자가 루프 전송을 더 많이 수행하면 §2.2에 정의 된 트랜잭션 그래프가 공격자의 순위가 낮아지도록 할 수 있다. 그 이유는 그러한 트랜잭션 그래프가 주화(Coinage) 및 권장 가치와 같은 요소를 고려하기 때문이다. 반면, Nebulas Rank는 이러한 요인을 강화할 수 있어 반(反)부정 기능성을 높였다.



(a) 루프 전송 횟수와 함께 공격자의 순위에 공격자의 자본 크기가 미치는 영향은 5,000으로 고정된다. (모든축은 대수 형태이다)



(b) 5.000으로 고정된 공격자의 자본과 공격자 순위에 대한 루프 전송 횟수의 영향 (모든 축은 대수 형태)

그림.5: 반(反)부정 기능

공격 방법은 **그림.4**와 같다. x축은 공격자의 자본을 나타내고, y 축은 공격자의 순위를 나타낸다(순위가 더 크다는 점은 공격자가 더 높은 점수를 얻지 못한다는 것을 의미한다. 이 점은 알고리즘의 높은 저항 능력을 나타낸다).

NR: 트랜잭션 그래프는 §2.2에 정의되어 있고 순위 알고리즘은 §2.3에 설명되어 있다.
NCD*: 트랜잭션 그래프는 NCDawareRank알고리즘과 함께 §2.2에서 정의된다.
NEM: 트랜잭션 그래프는 NCDawareRank와 함께 [38]에서 소개된다.
PR$^{\#}$: 트랜잭션 그래프는 페이지랭크 알고리즘과 함께 [38]에서 소개된다.
페이지랭크의 감쇠 계수는 0.15이다. NCDawareRank가 사용하는 클러스터링 알고리즘은 pscan[12], η = , μ = 0.1이다.

2.5 관련 연구

핵심 랭킹 지수인 중심성(Centrality)은 수십 년 전부터 네트워크 과학에서 가장 많이 연구된 개념이다[39]. 연결 중심성(degree centrality)[21], 고유 벡터 중심성(eigenvector centrality)[4], 카츠 중심성(Katz centrality) [27], 근접 중심성[50], 매개 중심성(betweenness centrality)[22][23][24][43][40], 페이지랭크[9], HITS[29], SALSA[51]를 포함하여 다양한 중심성을 소개하는 문헌이 있다. 게다가, 통일된 프레임워크[5][6][33]를 통해 이러한 척도를 명확히 분류하고 재검토하려는 몇 가지 기본 연구도 있다. Nebulas Rank 설계 시 적절한 중심성이 채택되기 전에 먼저 그래프의 속성을 고려해야 한다. 블록체인 트랜잭션 그래프의 시나리오는 [5]에서 언급한 자금 교환 흐름 네트워크와 가장 유사하다. 그러나 흐름 사이(flow betweenness) 중심성과 랜덤 워크(random-walk) 사이 중심성("current betweenness centrality"이라고 알려짐)[40]과 같은 작업에서 언급된 관련 알고리즘은 계산 집약적이며 대규모의 블록체인 트랜잭션 그래프이기 때문에 "계산 가능한" 특성을 만족시키지 못한다.

2009년에 발표된 비트코인[37] 시스템 이후로 연구진은 비트코인의 트랜잭션 그래프에 대한 통계적, 경험적 분석[49][26][41][2]을 수행했으며, 일부는 비트코인의 익명성을 논의하기 위해 트랜잭션 그래프 구조를 활용한다[34][44][46][20][19]. 다른 암호화폐가 등장하고 대중화된 후, 더 많은 블록체인에 대한 트랜잭션 그래프 분석이 수행된다[13][1]. Nebulas Rank는 사소한 것 몇 가지를 수정하여 트랜잭션 그래프 개념이자 주체 그래프(Entity Graph) [56]를 채택한다. 즉, 동일한 계정에 속한 각 계정 또는 계정 집합이 노드로 매핑되고 각 수신 엣지는 두 계정 간의 전송 강도를 나타낸다. 실제로 비트코인과 같은 블록체인 시스템이 발명되기 전 과학자들은 은행과 세계 무역 엔터티[48][8][52][3][17][36][7][30][53] 간의 금융 네트워크를 연구하려고 시도했다. 블록체인 트랜잭션 그래프와 비교할 때, 이러한 초기 연구 네트워크는 활동 이전뿐 아니라 대출과 같은 추가 정보로 정의된다. 또한 이러한 네트워크의 규모는 훨씬 작다. 결론적으로 대규모 트랜잭션 그래프, 특히 블록체인 트랜잭션 그래프에 대한 맞춤 순위 지정 방법을 제안하는 연구는 거의 없다.

Nebulas Rank와 가장 관련이 있는 연구는 NEM[38]의 중요도 증명 스키마이다. 네트워크 토폴로지의 클러스터링 효과를 활용하는 NCDawareRank[42]는 SCAN 알고리즘 [57][54][12]을 기반으로 한 클러스터링 알고리즘과 함께 순위 알고리즘으로 채택되었다. 커뮤니티 구조는 트랜잭션 그래프에 존재하면서 스팸 노드를 처리하는 데 도움이 되어야 한다. 하지만 커뮤니티 구조는 실제 세계의 한 주체가 제어하는 블록체인 세계의 모든 노드가 하나의 클러스터로 매핑되는 것을 보장하지는 않으므로 조작될 가능성으로 이어진다. 게다가, 플레더(Fleder), 케스터(Kester), 필라이(Pillai)[20]는 흥미로운 주소를 발견하고 그 주소의 활동을 분석하는 데 도움이 되는 지표로 페이지랭크[9][45]를 사용한다. 그러나 이들의 작업은 중요한 노드를 식별하는 자동화 프레임워크를 제공하지 않는다. 대신 Nebulas Rank의 맥락과 일치하지 않는 주관적인 분석에 의존한다. Nebulas가 선택한 알고리즘은 Leader Rank[14][31]이다. Learder Rank는 페이지랭크[9][45]의 단순하면서도 효과적으로 변형시킨 것이다. 페이지랭크에서 모든 노드에는 동일한 순간이동(teleportation) 매개 변수가 할당되는 반면 리더랭크는 그라운드 노드를 추가하여 각 노드에 다른 순간이동 매개변수를 할당한다. Nebulas Rank의 가중치 체계는 리(Li) 등의 영향을 받았다. [31]의 설계는 진입차수가 더 많은 노드에 이동으로 인한 방문 확률이 더 높다. Leader Rank 알고리즘을 채택하면 블록체인 시나리오에 더 적합한 결과를 얻을 수 있다.

3 Nebulas Force

Nebulas는 Nebulas Force(NF)를 사용하여 블록체인 시스템과 이와 관련한 어플리케이션의 진화 능력을 설명한다. 블록체인 시스템 및 어플리케이션 개발의 첫 번째 원동력인 Nebulas Force에는 Nebulas 가상 머신(Nebulas Virtual Machine, NVM), 블록체인 시스템의 프로토콜 코드 업그레이드, 블록체인 시스템에서 구동되는 스마트 컨트랙트 업그레이드라는 세 가지 측면이 있다.

Nebulas에서는 Nebuas 가상 머신(NVM)을 구현하기 위해 LLVM(Lower Level Virtual Machine)을 도입 할 것이다. 프로토콜 코드와 스마트 컨트랙트 코드는 NVM 바이트코드로 컴파일 된다. 이 코드는 LLVM(JIT/Just-In-Time) 적시 컴파일 기능으로 동적으로 컴파일 되고 최적화되며 최종적으로 샌드박스 환경에서 실행된다. 한편 LLVM의 모듈식 아키텍처를 통해 개발자는 익숙한 프로그래밍 언어를 사용하여 보다 안전하고 고성능의 스마트 컨트랙트를 구현할 수 있으며, 사용자에게 다양한 분산 어플리케이션을 제공할 수 있다.

Nebulas에서 프로토콜 코드를 업그레이드하기 위해 Nebulas는 체인의 추가 데이터를 보강할 것이며 프로토콜 코드 업그레이드를 수행용으로 블록 구조에 프로토콜 코드를 추가할 것이다. 이는 개발자와 커뮤니티 간의 분할 또는 분기를 피하기 위함이다. Nebulas 커뮤니티의 발전으로 Nebulas Force 및 기본 프로토콜의 업그레이드 기능이 점진적으로 커뮤니티에 공개되며, 커뮤니티는 Nebulas가 진화하는 방향을 규정하고 업그레이드 목표를 달성 할 것이다. Nebulas Force의 핵심 기술과 열린 개념 덕분에 Nebulas의 공간은 끊임없이 진화할 것이며, 그 가능성은 무한히 발전할 것이다. 예를 들어, 클라이언트 코드의 업그레이드가 존재하지 않는 대부분의 Nebulas 개발 과정에서, Nebulas Rank 알고리즘 매개 변수, PoD 인센티브 금액, 합의 알고리즘, 새 토큰 생성 속도를 포함한 일련의 매개 변수는 점진적으로 조정될 수 있다.

일반적으로 스마트 컨트랙트은 영구적인 것으로 간주되며 업그레이드가 지원되지 않는다. Nebulas는 상태 변수의 크로스 컨트랙트 방문(cross-contract visit)을 지원하는 스마트 컨트랙트의 기본 스토리지를 설계함으로써 스마트 컨트랙트의 업그레이드를 완료 할 수 있다. 이와 같은 해결책은 개발자에게 매우 친숙하므로 개발자가 버그에 보다 신속하게 대응할 수 있다. 따라서 해킹으로 인한 사용자의 엄청난 손실을 방지 할 수 있다.

3.1 Nebulas 가상 머신(NVM)

Nebulas는 LLVM[32]을 NVM의 주요 구성 요소로 LLVM 바이트코드를 NVM 바이트코드로 도입할 것이다. NVM 바이트코드는 LLVM JIT를 통해 동적으로 컴파일 되고 최적화되며 NVM 샌드박스 환경에서 실행된다. 이러한 아키텍처 설계를 통해 LLVM을 도입하여 Nebulas의 핵심 코드와 스마트 컨트랙트의 성능과 보안을 지속적으로 향상시킬 수 있다.

LLVM은 고도로 모듈화 된 컴파일러 툴체인 및 기술 모음으로, 구글(Google), 애플(Apple) 및 기타 여러 회사에서 코드 컴파일 프레임워크로 사용되었다. LLVM은 중립적인 중간 표현(LLVM IR)과 해당 컴파일 인프라를 제공하며 이러한 인프라와 관련된 새로운 컴파일 전략을 제공한다. **그림. 6**에서 볼 수 있는 LLVM IR에서 LLVM 바이트코드로의 코드 생성, LLVM JIT를 통한 다른 하드웨어 플랫폼에서의 LLVM 바이트코드 직접 실행이 포함된다.

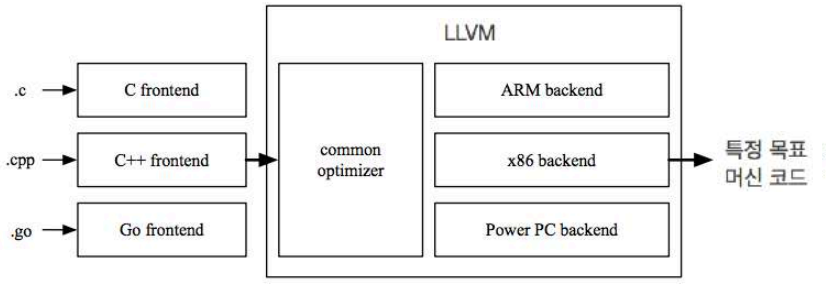


그림. 6: LLVM

Nebulas는 **그림.7**에서와 같이 LLVM에 기반한 NVM을 구축한다. 우선, 블록체인용 기본 API 라이브러리를 제공한다. 그리고 솔리디티(Solidity), 자바스크립트(JavaScript), C / C ++, Go 등 다른 언어에 대한 컴파일러 프론트 엔드를 작성한다. 그 후, LLVM에서 제공하는 툴체인을 사용하여 LLVM 바이트코드를 생성한다. 마지막으로, LLVM 바이트코드는 LLVM의 JIT 엔진을 통해 NVM이 제공하는 안전한 샌드박스 환경에서 실행된다.

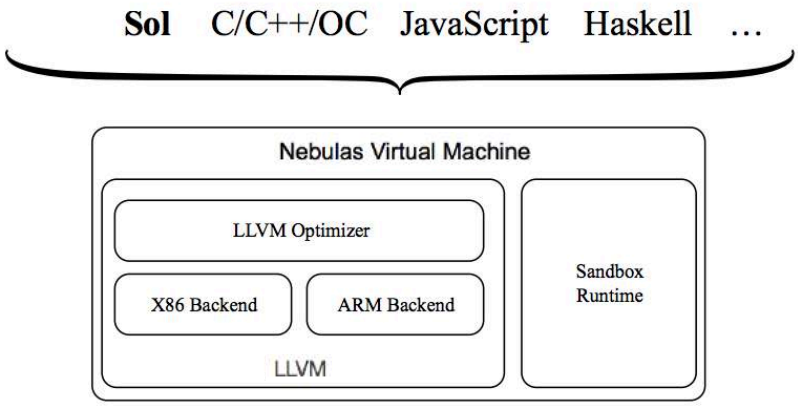


그림. 7: Nebulas 가상 머신(NVM)

NVM은 Nebulas Force의 중요한 초석이다. 새로운 프로토콜 코드 또는 스마트 컨트랙트가 공개되면 NVM의 LLVM 컴파일러 모듈이 새 코드를 컴파일하고, 새 코드가 체인에 배포 된 후 LLVM 바이트코드가 생성된다. 체인에서 확인된 후 LLVM JIT에서 새 코드가 컴파일되고 최적화 된 다음, 기존 코드를 교체하여 샌드박스에서 실행된다(그림 8 참조).

LLVM(그림 6 참조)을 통해 NVM은 개발자가 솔리디티(Solidity), 보다 유연한 자바스크립트(JavaScript), 순수 함수형 언어인 하스켈(Haskell)과 같은 익숙한 프로그래밍 언어로 스마트 컨트랙트 및 어플리케이션을 개발할 수 있도록 지원한다. 이러한 대중적인 언어 외에도 NVM은 DSL(도메인 특화 언어/domain-specific language)과 같은 다양한 영역 및 시나리오에 맞는 맞춤형 고급 언어를 제공 할 수 있다. 이러한 고급 언어는 공식적으로 검증하기 쉽고, 코드 견고성 및 보안을 향상시키며, 풍부한 스마트 컨트랙트 및 어플리케이션을 개발하는 개발자에게 보다 도움이 된다.

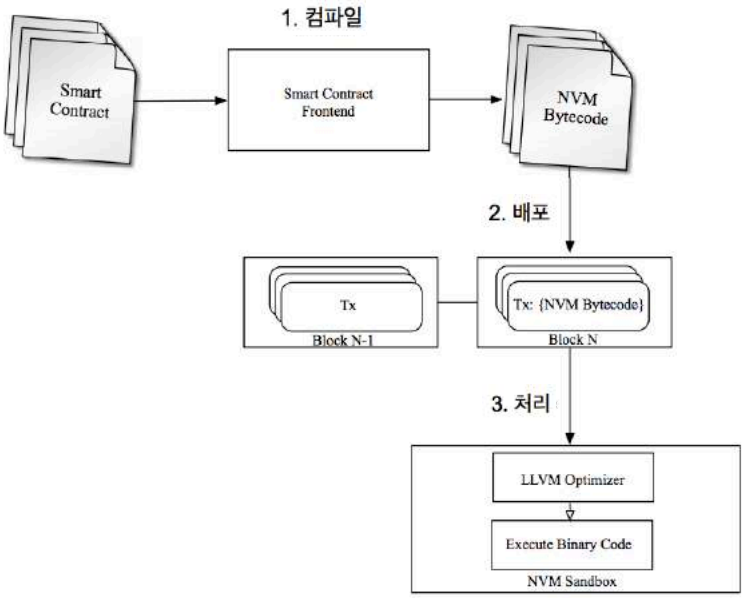


그림. 8: Nebulas 가상머신(NVM)의 작동 매커니즘

3.2 프로토콜 코드의 업그레이드 설계

우선 Nebulas의 블록 구조를 제공한 다음 블록 구조를 기반으로 프로토콜 코드를 업그레이드하는 방법에 대해 논의한다.

3.2.1 블록 구조

Nebulas 블록 데이터 구조는 다음을 포함하지만 이에 국한되지 않는다.

- 헤더(Header) : 블록 헤더(Block Header)
  - 높이 : 블록 높이
  - 부모해시(ParentHash) : 부모 블록 해시(parent block hash)
  - Ts : 타임스탬프(TimeStamp)
  - 채굴자(Miner) : 채굴자 주소(miner address)
  - 다이너스티(Dynasty) : 블록의 합의 다이너스티
  - 에폭(Epoch): 블록의 합의 연령
  - 스테이트루트(StateRoot) : 스테이트 루트 해시(state root hash)
  - TxsRoot : 트랜잭션 루트 해시(transaction root hash)
  - ReceiptsRoot : 트랜잭션 수령 해시(transaction receipt hash)
  - TransNum : 트랜잭션 수
- 트랜잭션 : 트랜잭션 데이터 (여러 거래 포함)
  - From : 트랜잭션 발신자 주소
  - To : 트랜잭션 수신자 주소, 가치 0x0의 스마트 컨트랙트 생성용가치 : 트랜잭션 금액
  - 데이터 : 트랜잭션 페이로드. 트랜잭션이 스마트 컨트랙트를 생성하는 경우 스마트 컨트랙트 바이트코드, 트랜잭션이 스마트 컨트랙트 콜인 경우 호출 함수 및 항목의 이름이다.
  - 서명 : 트랜잭션 서명
  - 가스(Gas) : 가스 한도(gas limit)
  - 가스가격(GasPrice) : 가스 단가(gas unit price)
  - 논스(Nonce) : 트랜잭션의 고유성
- 표 : 프리페어(Prepare) 표 와 커밋(Commit) 표 and Commit Votes (다중 포함), PoD (see §5) 합의 알고리즘에서 사용
  - From : 투표자
  - VoteHash : 표를 받은 블록의 해시
  - Hv : 표를 받은 블록의 높이
  - Hvs : 표를 받은 블록의 조상 블록의 높이
  - VoteType : 투표 타입 , 프리페어 또는 커밋
  - 서명 : 투표 서명
- 프로토콜 코드 : 프로토콜 코드(블록에서 0 또는 1만)
  - Hash : 프로토콜 코드 해시
  - Code : 프로토콜 코드의 바이트코드
  - ValidStartBlock : 프로토콜이 효력을 발생하는 시작 블록 번호
  - 서명 : 서명(개발자 커뮤니티가 서명함)

– 버전(Version) : 프로토콜 코드 버전 번호. 악의적 인 계정이 이전 프로토콜 코드로 롤백되지 않도록 각 업그레이드가 증가되어야 한다.

– 논스(Nonce) : 프로토콜 코드 독창성

- Nebulas Rank : Nebulas 인덱스(일주일에 1회 계산되며, 대부분의 블록은 이 섹션이 없다.)
  – 랭크버전(RankVersion) : NR 버전
  – 랭크루트(RankRoot) : NR 랭크 해시
  – 랭크레코드(RankRecords) : NR 랭크 레코드
  * 주소(Address) : 계정 주소 태그
  * 점수(Score) : Nebulas Rank 가치

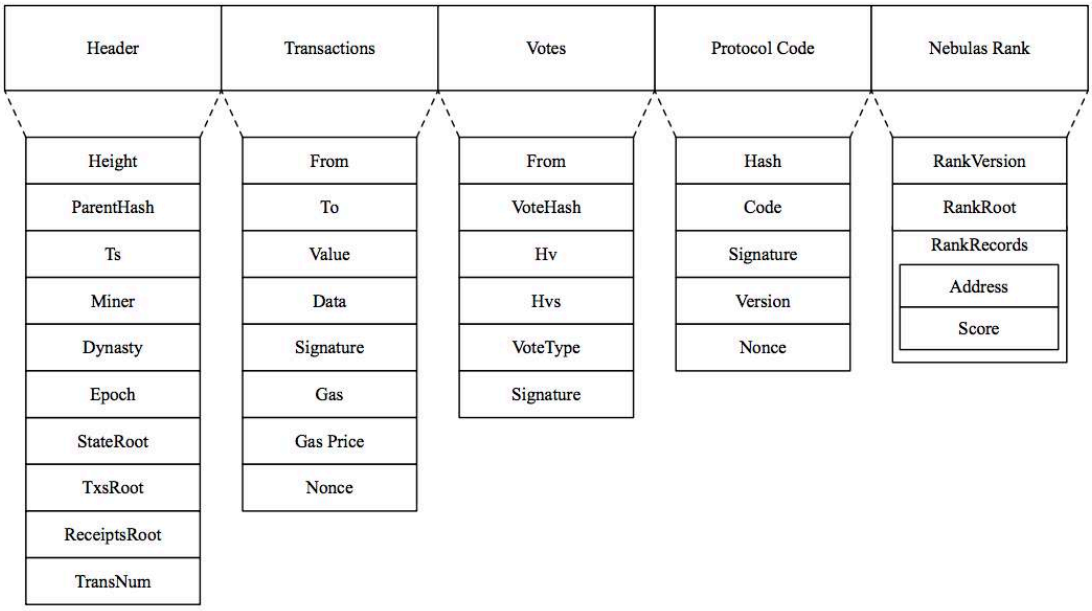


그림. 9: 블록 구조

다른 암호화폐 시스템과 마찬가지로 계정과 블록체인 간의 상호 작용은 특정 트랜잭션을 통해 수행된다. 이러한 계정은 개인 키로 서명 된 트랜잭션을 생성하고 이를 블록체인의 모든 노드로 보내고 P2P 네트워크를 통해 전체 네트워크 노드로 유포한다. 고정 블록 시간 간격 동안 PoD 합의 알고리즘 (§5참조)에 의해 지정된 노드는 해당 시간의 모든 트랜잭션을 수집하여 이를 표준 형식 블록으로 묶어 나머지 네트워크로 배포한다. 각 노드에서 확인된 후 새 블록이 로컬 원장에 추가된 후, 글로벌 원장에 포함된다.

이더리움에서 트랜잭션은 보통 계정 트랜잭션 및 스마트 컨트랙트 트랜잭션이라는 두 가지 유형으로 나뉜다. 우리는 Nebulas 블록에 새로운 트랜잭션 유형인 프로토콜 코드 및 네뷸러스 랭크를 추가한다. 블록체인 데이터의 일부인 프로토콜 코드는 체인에 저장되고 네뷸러스의 기본 프로토콜 업그레이드는 체인에 대한 추가 데이터를 보완하여 수행된다. Nebulas Rank의 랭킹 알고리즘에 기반하여 각 계정의 Nebulas Rank 값은 각 사이클에서 계산되어 실시간 Nebulas Rank 값 호출 및 히스토리 순위 쿼리를 돕기 위해 해당 체인에 저장된다.

3.2.2 프로토콜 코드 업그레이드

Nebulas 클라이언트 노드는 최신 블록의 프로토콜 코드 저장소 영역에서 컴파일 된 가상 머신 바이트코드 (NVM 바이트코드)를 획득 할 수 있다. 최신 블록에 프로토콜 코드 데이터가 없으면 프로토콜 코드가 변경되지 않았다는 사실을 나타내며 가장 가까운 블록의 프로토콜 코드로 역 추적해야 한다. 블록체인 프로토콜 코드의 모든 동작은 인증 알고리즘, 패킹 규칙, Nebulas Rank 알고리즘, 인센티브 매커니즘 등을 포함하여 프로토콜 코드에 의해 결정된다. 블록체인의 거의 모든 동작은 프로토콜 코드에 의해 정의 될 수 있다.

프로토콜 코드를 업그레이드해야 하는 경우 Nebulas 개발 팀이 개발을 담당하며 커뮤니티에서 토론 및 투표를 위한 채널을 열 수 있는 코드가 공개된다. 투표는 스마트 컨트랙트 또는 포럼에서 투표의 형태로 수행 될 수 있다. 대다수의 커뮤니티 회원이 프로토콜 업그레이드에 동의 시 Nebulas 개발 팀은 최신 코드를 프로토콜 코드 트랜잭션에 포함시켜 전체 네트워크의 모든 노드에 배포한다. 북키핑 노드(bookkeeping nodes)가 이를 블록에 포함하는 경우에만 지정된 블록 높이에서 유효하게 된다. 이러한 유형의 블록체인 프로토콜 업그레이드는 소프트 포크 및 하드 포크가 없는 고객에게도 투명하게 진행된다.

승인 후에 프로토콜 코드가 공개되도록 하기 위해 프로토콜 코드 게시자는 핵심 Nebulas가 보유한 주소를 사용하고 최초 블록 내에서 하드 코드로 변경할 수 없다. 모든 북키핑 노드는 프로토콜 코드의 서명을 검증할 것이다. 서명이 검증을 통과하지 못하면 불법 데이터로 간주된다.

후속 개선 조치는 프로토콜 코드의 서명 검증을 N 중의 M 다중 서명(M-of-N multi-signature)으로 변경하는 것으로, 이는 프로토콜 코드 업그레이드를 통해 구현할 수도 있다.

3.3 스마트 컨트랙트의 업그레이드 설계

3.3.1 튜링 완전 스마트 컨트랙트 프로그래밍 언어

스마트 컨트랙트는 컨트랙트 참여자가 약속을 이행하기 위한 컨트랙트와 같이 디지털 형식으로 정의된 일련의 약속이다. 실제로 스마트 컨트랙트의 매개체는 컴퓨터가 인식하고 컴퓨터에서 작동할 수 있는 컴퓨터 코드이다. 비트코인 스크립트 언어는 필수적인 스택 기반의 프로그래밍 언어이며, 불완전한 튜링이기 때문에 어플리케이션에 몇 가지 제한이 있다. 이더리움은 튜링 완전 스마트 컨트랙트를 구현한 세계 최초의 블록체인 시스템이다. 채택된 프로그래밍 언어는 솔리디티(Solidity)와 서펜트(Serpent)이다. 이를 통해 개발자는 빠르고 다양한 방법으로 다양한 응용 프로그램을 개발할 수 있다. 스마트 컨트랙트 코드가 블록체인에 게시된 후에는 대리인의 참여 없이 자동으로 실행될 수 있다.

초기 단계에서 Nebulas의 스마트 컨트랙트 프로그래밍 언어는 이더리움의 솔리디티(Solidity)와 완벽하게 호환되어 개발자는 이더리움용으로 개발 된 스마트 컨트랙트 어플리케이션을 Nebulas로 원활하게 이전 할 수 있도록 했다. 우리는 Nebulas Rank와 관련된 몇 가지 명령 집합을 솔리디티(Solidity) 언어에 추가하여 개발자가 모든 사용자의 Nebulas Rank 가치를 얻을 수 있도록 한다. NVM을 기반으로 다양한 프로그래밍 언어를 지원하므로 개발자는 자바(Java), 파이썬(Python), 고(Go), 자바스크립트(JavaScript), 스칼라(Scala) 등과 같이 자신이 선호하는 고급 언어로 프로그래밍하거나 특정 분야의 고급 언어로 특정 어플리케이션을 쉽게 만들 수 있다.

3.3.2 업그레이드 가능한 컨트랙트

현재, 이더리움 스마트 컨트랙트 설계용 코드가 일단 작성되면 더 이상 변경할 수 없다. 코드 로직이 마련되는 순간부터 영원히 업그레이드 할 수 없다. 스마트 컨트랙트가 합의 역할을 하는 경우 변경이 되어서는 안 된다. 이는 연산이 확정된 합의를 나타낸다. 그러나 스마트 컨트랙트가 점점 더 널리 사용됨에 따라 워크 플로우와 코드가 점점 더 복잡해졌다. 그리고 스마트 컨트랙트는 현실 세계에서 실제 계약처럼 보인다. 신중하게 검토하지 못할 경우 설계 및 코딩 프로세스에서 인적 오류를 예방하는 것이 어렵다. 해커가 버그를 발견하면, 손실은 항상 매우 심각하다. 2016년 6월, DAO 공격은 코드 결함으로 인해 초래되었고, 이더리움 사용자에게 총 6,000만 달러의 손실을 입혔다. 또한 패리티 지갑(Parity Wallet)의 최근 버그로 인해 3천만 달러에 준하는 15만 ETH의 손실이 발생했다. 비트코인은 튜링의 불완전성으로 설계되었고 많은 스크립트 지침이 삭제 되었기 때문에, 비트코인의 보안 수준이 매우 높다.

스마트 컨트랙트 프로그래밍의 모범사례도 존재하고, 수학적 증명을 통한 스마트 컨트랙트의 확실성을 확인하는 더 엄격한 검토 프로세스 및 공식 검증 도구가 존재한다. 하지만 버그가 전혀 존재하지 않는 코드는 없다고 볼 수 있다. 현재 중앙 집중식 인터넷 세계를 살펴보면, 인터넷 서비스를 업그레이드하여 개발 프로세스에서 나타나는 여러 가지 버그를 해결할 수 있다. 완벽한 어플리케이션 시스템을 설계할 수는 없지만, 완벽한 시스템을 향해 점진적으로 발전할 수는 있다. 스마트 컨트랙트의 보안 문제를 분류하는 기본적인 요건은 스마트 컨트랙트의 업그레이드가 가능한 설계 솔루션을 만들어 내는 것이다.

이더리움에서 스마트 컨트랙트를 업그레이드 할 수 있는 설계에는 몇 가지 솔루션이 있으며 일반적으로 두 가지 범주로 나뉜다. 첫 번째는 대중이 사용할 수 있는 프록시 컨트랙트(Proxy Contract)다. 프록시 컨트랙트 코드는

```
[baseContractAddress="0x5d65d971895edc438f465c17db6992698a52318d"]
//baseContractAddress is the address of the old contract
contract Token {
  mapping (address => uint256) balances shared;

  function transfer(address _to, uint256 _value) returns (bool success) {
     if (balances[msg.sender] >= _value && _value > 0) {
       balances[msg.sender] -= _value;
       balances[_to] += _value;
       return true;
     } else {
       return false;
     }
   }
   function balanceOf(address _owner) constant returns (uint256 balance) {
       return balances[_owner];
   }
}
```

```
       return false;
     }
   }
   function balanceOf(address _owner) constant returns (uint256 balance) {
       return balances[_owner];
   }
}
```

그림. 10: 컨트랙트 코드 원본

컨트랙트가 배포되면 균형 변수(balances variable)는 공유(shared) 키워드로 표시되고, 원본 컨트랙트 코드는 작업을 위해 바이트코드로 컴파일 되며, 가상 머신은 이 변수에 대해 별도로 저장 영역을 설계한다. 공유 키워드로 표시되지 않은 변수의 경우 다른 컨트랙트에서 원본 계약 코드를 직접 접속 할 수 없다.

원본 코드의 전송 함수에서 버그가 수정되어야 하는 경우 **그림.11**에 표시된 대로 _value를 확인하고 새로운 스마트 컨트랙트 코드를 배포한다.

신규 컨트랙트가 배포 된 후, 기존 컨트랙트는 자기파괴를 선택할 수 있다. 즉, 더 이상 접속 할 수 없다. 그러나 공유 변수는 영구적으로 보존된다. 신규 컨트랙트는 상태를 잃지 않은 상태에서 기존 컨트랙트 잔액 자산을 완전히 물려받을 수 있으므로 추가 이전이 필요하지 않다. 그러나 스마트 컨트랙트를 개발할 때 중요한 상태 변수를 공유하도록 선언해야 한다. 컴파일러는 변수의 저장 영역을 특별히 처리하여 인가된 기타 컨트랙트가 접속할 수 있도록 보장한다.보안을 유지하기 위해 컨트랙트 업그레이드와 이전 컨트랙트는 동일한 작성자를 이용해야 하며, 그렇지 않으면 연산 중 예외가 발생한다.해당 설계에는 도덕적인 문제가 있다. 일단 컨트랙트 조항이 제안되고 체결되면 수정이 되어서는 안 된다. 모든 변경 사항에 대해 컨트랙트 대상자 동의해야 한다. Nebulas는 스마트 컨트랙트의 업그레이드를 승인하는 투표 메커니즘을 도입하여 컨트랙트 작성자가 컨트랙트를 자동으로 수정하는 행위를 방지한다.

그림. 11: 신규 컨트랙트 코드

이 업그레이드 가능 솔루션을 사용하면 DAO 공격 또는 패리티 버그 또는 유사한 버그 공격 사건이 발생했을 경우, 하드 포크를 진행하지 않으며 더 신속하게 수정 될 수 있다. 또한, 수정 후 모든 사용자의 자산을 이전하지 않고 지속적으로 사용할 수 있다.

4 Developer Incentive Protocol

4.1 설계 목표

Nebulas를 위한 양질의 커뮤니티를 조성하기 위해 스마트 컨트랙트 개발자용 Developer Incentive Protocol(DIP, 개발자 인센티브 프로토콜)을 제공하고 Nebulas 블록체인과 생태계에 기여하는 모든 개발자에게 그에 합당한 인센티브(NAS)로 보답하고자 한다.

4.2 DIP 인센티브 지급 알고리즘

우수한 스마트 컨트랙트의 기준은 얼마나 많은 사용자가 사용하는지에 따라 달려 있다. 가치가 높은 계정이 많으면 스마트 컨트랙트가 늘어 난다. 보편적 가치 평가 기준의 일환으로, Nebulas Rank은 가치가 높은 계정을 평가하는 데 사용될 수 있다. Develper Incentive Protocol의 설계는 Nebulas Rank과 일반적인 WAU(Weekly Active Users, 주간 활성 사용자) 개념을 결합한 것으로 전체 WAU 값은 스마트 컨트랙트의 가치 측정을 평가하는 데 사용된다.

Developer Incentive Protocol는 일주일에 한 번 실시된다. 스마트 컨트랙트 C의 경우 이번 주의 활성 계정 주소 집합이 WAA (주간 활성 주소/Weekly Active Addresses)라고 가정한다. §2.3의 Nebulas Rank의 랭킹(상위 X 포함)에 따르면 주간 활성 주소의 Nebulas Rank의 합은 **방정식.9**와 같이 계약 C의 SCS(스마트 컨트랙트 점수/Smart Contract Score)로 계산된다.

$$SCS(C) = \sum_{addr \in WAA} (max\{X + 1 - NR(addr), 0\}) \tag{9}$$

이 값은 매주 SCS를 기준으로 SCR(스마트 컨트랙트 순위/Smart Contract Rank)에 대해 높은 점수에서 낮은 점수로 분류된다. 상위 N개의 스마트 컨트랙트가 포함되며, 해당 개발자는 비율에 따라 M개의 NAS를 공유한다. 악의적인 랭킹 사기를 피하기 위해 Developer Incentive Protocol 분포 곡선은 **그림.12**와 같이 고르도록 설계되었지만 SCS의 차이를 나타내기 위해 순위 1의 수익이 순위 N의 2배인 것을 여전히 보장한다. 비율 제약 조건은 **방정식.10**과 같다.

$$\begin{aligned} Coin(C) = \quad & kln(N + 1 - SCR(C)) + b \\ \text{s.t.} \quad & kln(N) + b = 2b \\ & \sum_{x=1}^{N} (kln(x) + b) = M \end{aligned} \tag{10}$$

Developer Incentive Protocol 보상은 별도로 계산되어 각 노드별로 분배된다. Nebulas의 한 블록이 S초마다 생성된다고 가정하면 Developer Incentive Protocol 보상은 모든 노드에 대해 24*7*3600 / S 블록마다 한 번 계산되고 해당 스마트 컨트랙트의 현금 지급 주소로 분산된다.

Nebulas 생태 스마트 컨트랙트의 다양성을 장려하고 더 많은 신규 개발자가 뛰어난 결과를 내도록 장려하기 위해 Developer Incentive Protocol는 각 스마트 컨트랙트가 K시간까지 보상받을 수 있다고 규정한다.

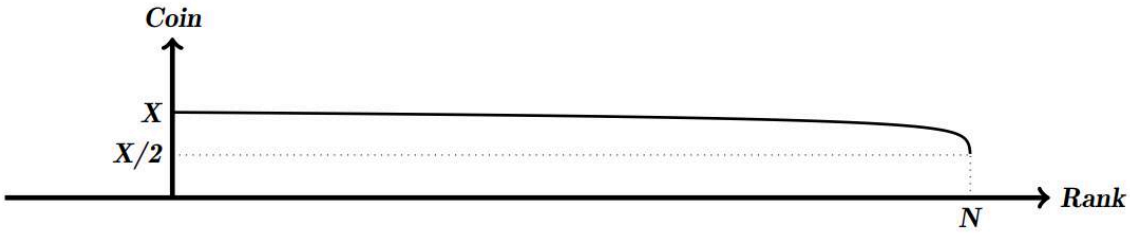


그림. 12: Developer Incentive Protocol 보상 지급 곡선

Developer Incentive Protocol는 순위에 따라 보상 자격이 부여된 상위 N 스마트 컨트랙트를 선택하여 블록체인 어플리케이션의 생태계 구축을 위한 개발을 촉진한다.

4.3 실험 결과

Nebulas에서는 이더리움에서 2017년 5월에 발생한 트랜잭션 데이터를 수집하고 **표.2**에 표시된 대로 1주차 Developer Incentive Protocol 순위를 계산했다.

표.2: 2017년 5월[3] 1주차 Developer Incentive Protocol Top 10 랭킹 결과

컨트랙트 주소	점수	설명[4]
0xa74476443119a942de498590fe1f2454d7d4ac0d	264456363.0	GolemToken
0x49edf201c1e139282643d5e7c6fb0c7219ad1db7	207900181.0	TokenCard-ICO
0x48c80f1f4d53d5951e5d5438b54cba84f29f32a5	129625776.0	REP-Augur-OLD
0x6810e776880c02933d47db1b9fc05908e5386b96	108324489.0	Gnosis-TokenContract
0x6090a6e47849629b7245dfa1ca21d94cd15878ef	54429341.0	ENS-Registrar
0x607f4c5bb672230e8672085532f7e901544a7375	48526808.0	RLC
0x8d12a197cb00d4747a1fe03395095ce2a5cc6819	46498412.0	etherdelta_2
0xf7b098298f7c69fc14610bf71d5e02c60792894c	43746158.0	GUPToken
0xaaaf91d9b90df800df4f55c205fd6989c977e73a	42026627.0	TokenCardContract
0xaec2e87e0a235266d9c5adc9deb4b2e29b54d009	41427314.0	singularDTVToken

최상위 순위의 컨트랙트는 랭킹 산출주기에서 기여도가 높고 더욱 활동적이다. 이는 본래 에코빌더(eco-builder)를 장려하기 위한 의도와 일치한다는 점을 알 수 있다.

4.4 부정행위 분석

스마트 컨트랙트는 수동적으로만 호출이 가능하다. 따라서 부정행위자(cheater)가 스마트 컨트랙트에서의 순위를 높이려면 자신의 계약을 호출하기 위해 높은 등급의 Nebulas Rank 계정를 찾아야 한다.

우선, 부정행위자가 비용을 전혀 들이지 않고 Developer Incentive Protocol의 순위를 조작하는 것은 불가능하다. 부정행위자가 컨트랙트 C(Contract C)의 순위를 높이길 원하고, 그 목적으로 많은 수의 계정를 위조한다고 가정해보자. 그러나, SCS가 **방정식.9**에서 계산될 때, Nebulas Rank의 랭킹에서 SCS 호출의 상위 X만이 0보다 크지만, 새로 위조된 계정의 Nebulas Rank의 랭킹은 상위 X 밖에 있다. 컨트랙트 C가 호출 되더라도, Developer Incentive Protocol 순위에 영향을 미치지 않는다.

[3] 블록 범위 [3629091, 3665815]

[4] 출처: etherscan.io

둘째, 부정행위자가 컨트랙트의 Developer Incentive Protocol의 순위를 향상시키기 위해 특정 비용을 지불하려는 경우 두 가지 옵션을 사용할 수 있다. 첫 번째는 Nebulas Rank에서의 랭크가 높은 계정을 위조하고 컨트랙트 C의 순위를 향상시키기 위해 컨트랙트 C를 자비로 호출하는 것이다. §2.4의 높은 랭킹을 가진 위조 계정에 대해 분석되어있다. 각 계정에 대한 랭킹을 향상 시키려면 이러한 계정의 특정 토폴로지를 위조하기 위해 엄청난 비용이 필요하다. 게다가, Nebulas Rank은 주기적으로 갱신되므로, 높은 랭킹을 유지하는데에도 상당한 비용이 든다. 두 번째는 Nebulas Rrank에서의 랭킹이 높은 많은 수의 계정를 찾고 소유자에게 컨트랙트 C를 하도록 설득하거나 뇌물을 제공하는 것이다. 그러나 이러한 오프 체인(off-chain) 작업은 확장성에 있어서 매우 제한적이며 어렵다. 더군다나 상당한 비용을 들여 부정행위자가 찾은 랭킹이 높은 계정은 상위 X의 일부에 불과하다. 실제로 높은 랭킹에 위치한 컨트랙트에 거의 영향을 미치지 않는다.

5 Proof of Devotion(PoD) 합의 알고리즘

5.1 설계 목표

합의 알고리즘은 블록체인의 초석이며, 신속성과 비가역성이 Nebulas의 주안점이다. Nebulas 블록체인 상의 보다 나은 생태계를 구축하기 위해 공정성 또한 매우 중요하게 생각한다. 대규모의 자본을 통해 Nebulas의 블록 합의 알고리즘을 쉽게 통제할 수 있다면, 많은 개발자와 사용자의 이익이 손상될 것이다. 또한, 기여자의 이익을 보장할 수 없는 생태계는 보다 고차원적인 가치를 창출하는 것이 어렵다. 따라서 합의 알고리즘은 Nebulas 생태계를 위해 기여하는 모든 이들의 이익을 보장하기 위해 가능한 공정성을 추구하는 기반 위에서 신속성과 비가역성을 보장하도록 설계되어야 한다.

5.2 널리 사용되는 합의 알고리즘의 결함

Nebulas는 널리 사용되는 합의 알고리즘 중 설계 목표와 일치하는 것을 찾기 위해 노력했지만 이러한 알고리즘은 양질의 Nebulas 생태계를 위한 조건을 완벽하게 충족시킬 수 없었다.

PoW(Proof of Work, 작업증명) 합의 알고리즘은 제로섬 게임으로, 경쟁력 있는 해시 연산을 활용하여 북키퍼를 결정하고, 블록 생성을 위한 경쟁에서 많은 양의 전력을 낭비한다. 따라서 채굴 비용이 높고 속도도 제한적이다. 채굴을 위한 노드의 증가로 인해 각 노드가 북키핑(bookkeeping) 권한을 얻을 확률이 줄어들어 PoW 프로토콜에 따라 블록을 안정적으로 생성하는데 드는 비용이 지속적으로 증가한다. 이더리움은 오랫동안 현재의 PoW 합의 알고리즘[11]을 새로운 PoS(지분 증명) 합의 알고리즘인 캐스퍼(Casper)[55]을 채택함으로써 점차적으로 기존의 합의 알고리즘을 대체하는 것을 고려해 왔지만 채굴의 어려움을 계속 증가시키는 비트코인의 경우 채굴 머신의 수지 타산이 맞지 않는 상황에 조만간 직면하게 될 것이다. 채굴 속도와 경제적 비용 측면에서 PoW(작업 증명)는 Nebulas의 생태계의 장기적이며 지속적인 발전에 도움이 되지 않는다는 것을 알 수 있다. Nebulas 합의 알고리즘의 “신속성” 반하는 점이다.

PoS(지분 증명) 합의 알고리즘은 자본을 사용하여 해시 비율을 대체하고 금액에 따라 북키핑 권한을 획득할 확률을 분배하려고 하는 것이다. 현재 피어코인(Peercoin)[28]과 이더리움의 캐스퍼(Casper) 프로토콜은 PoS(지분 증명) 합의 알고리즘을 채택하고 있다. 이 알고리즘은 PoW의 높은 전력 소비라는 단점을 극복하고 자본이 북키핑이 될 수 있는 기회에 미치는 영향을 시각적으로 확대한다. PoW와 비교하면, PoS의 대규모 자본은 생태계를 통제하고 집단 독점을 형성할 가능성이 크기 때문에 Nebulas 생태계에 기여하는 모든 이들의 이익에 반하고 Nebulas의 가치 창출에 악영향을 미칠 수 있다고 판단하였다. 이 모든 점은 Nebulas 합의 알고리즘의 “공정성”과 어긋난다.

PoI(Proof of Importance, 중요도증명) 합의 알고리즘은 Nem[38]에서 처음 제안했다. PoS와는 달리, 계정 중요성 개념이 PoI에 도입되고 계정 중요도 점수가 사용되어 북키퍼가 될 확률을 안배한다.

PoI 합의 알고리즘은 PoW의 높은 전력 소비 단점을 극복하고 PoS 자본 주의식 독점을 다소 해결할 수 있지만 Nothing at Stake이라는 문제를 직면하게 된다. 부정행위자가 블록을 뒤바꿀 때 드는 비용이 크게 줄어들어 Nebulas 합의 알고리즘의 “비가역성"과 어긋난다.

즉, 널리 사용되는 합의 알고리즘과 이에 대한 Nebulas의 목표가 불일치함으로, 포괄적으로 계정의 영향력과 기여도를 평가하는 PoI와 엄격한 경제적 패널티를 포함하는 PoS를 결합한 새로운 PoD(기여도 증명) 알고리즘을 개발하고자 한다. PoS는 PoI의 비가역성을 향상시키고, PoI는 PoS의 독점을 예방할 수 있다.

5.3 PoD 합의 알고리즘의 설계

5.3.1 새로운 블록의 생성

중요도가 높은 순위에 따라 계정을 선정하는 PoI 합의 알고리즘과 유사하지만, PoD는 생태계에 대한 영향력과 기여도에 따라 계정을 선정한다. 또한 한쪽으로만 치우쳐진 권한을 막기위해, PoD를 통해 선정된 계정은 북키핑의 권한을 부여받으며 새로운 블록 생성을 위한 참여 기회도 똑같이 주어진다. 영향력과 기여도가 큰 계정을 선정 시 Nebulas Rank를 사용한다. Nebulas Rank의 알고리즘 설계에서 계정의 유동성과 전파성을 강조했다 (§2.1 참조). 이러한 속성을 포함한 계정이 Nebulas의 생태계 구축과 관련하여 높은 영향력과 기여도를 가질 것이라고

할 수 있다. 따라서 PoD를 통해 Nebulas Rank의 상위 N 계정이 선정되며 이 계정은 자발적으로 일정 금액(NAS)를 보증금으로 입금하면 북키핑에 참여할 시 새로운 블록 생성에 대한 검증자 자격이 부여된다.

여러 검증자가 선정되어 무리가 이루어지면 PoD 알고리즘은 의사 무작위(pseudo-random) 숫자를 사용하여 검증자 무리 중 누가 새로운 블록 생성자인지, 어떤 트랜잭션을 압축하여 새 블록을 생성해야 하는지 결정한다. 검증자는 바뀔 수 있으며 검증자 조건을 충족시키는 계정은 검증자로서의 권한을 요청하거나 또는 권한을 취소요청 할 수 있다. 또한 검증자 조건을 충족시키는 계정은 Nebulas Rank의 정기적인 변경사항에 따라 달라질 수 있다. 따라서 Nebulas는 검증자 무리의 교체를 구현하기 위해 PoD를 기반으로 하는 검증자 무리 교체 매커니즘을 설계했다.

5.3.2 검증자 무리 교체 매커니즘

검증자 무리는 한 Dynasty(시대)의 방식으로 교체되므로 무리별로 다른 Dynasty로 나뉘며, 한 시대 내의 검증자 무리는 교체되지 않는다. 한 Dynasty 안에서 급격한 교체는 있을 수 없으며 일정 기간 내에 교체되지 않아야 한다. 따라서 Nebulas는 모든 X 블록을 특정 Epoch(세대)로 정하고, 동일한 Epoch에서는 어떤 교체도 일어나지 않는다. 그러므로 Dynasty의 변화는 하나의 Epoch에서 다른 Epoch로 넘어갈 때에만 발생한다. Epoch 교체가 일어날 때 이전 Epoch의 첫 번째 블록은 조사를 받게 된다. 첫 번째 블록이 Finality(최종) 상태에 돌입한다면 현재 Epoch가 다음 D1 Dynasty에 진입하게 된다. 만일 조사가 이루어지지 않는다면 기존의 D0 Dynasty가 그대로 유지된다. Dynasty 교체에 대한 과정은 **그림.13**에서 추가적으로 설명한다.

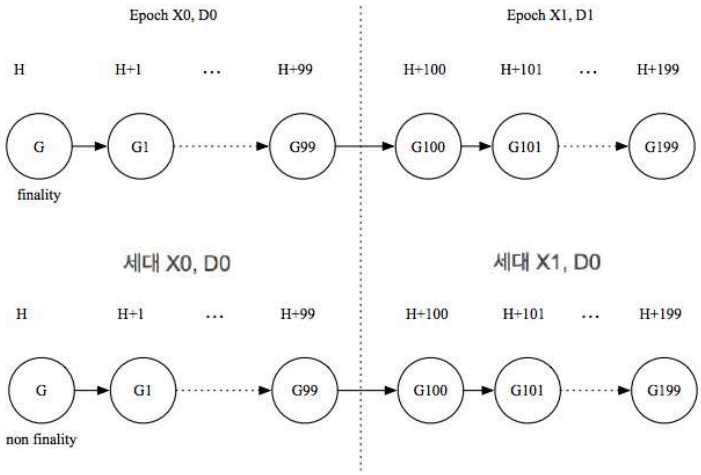


그림. 13: 검증자 무리의 교체(X=100으로 가정)

Dynasty 변경이 발생할 시, 네트워크 지연으로 인해 각 노드에서 블록 G의 Finality(최종) 상태가 동일하지 않을 수 있다. 검증자 무리 교체 매커니즘을 기반으로 각 Dynasty(시대)의 합의 과정은 현재와 이전 Dynasty의 검증자 무리가 함께 결정한다. 따라서 어떤 Dynasty에서도 적절한 계정은 Dynasty D+2의 검증자 무리에 합류하거나 해지 신청만 진행할 수 있으며, Dynasty가 Dynasty D+2로 진화하면 새로운 블록의 합의 과정에 참여할 수 있게 된다.

5.3.3 합의 과정

새로운 블록이 제안된 후 현재 Dynasty의 검증자 무리에 있는 모든 이들은 현재 블록의 정당성을 결정하기 위해 BFT 스타일(Byzantine-Fault-Tolerant Style, 비잔틴 장애 허용 스타일) 투표에 참여한다. 투표 시작 시, 해당 블록 합의에 참여하는 각 검증자는 보증금으로 2x(x는 인센티브 보너스 비율)가 청구되고 두 단계의 투표 과정이 시작된다.

- **첫 번째 단계:** 모든 검증자는 새 블록에 대하여 $Prepare$ tickets(프리페어 티켓)을 제안해야 한다. $Prepare$ tickets을 제안한 후, 검증자는 1.5배의 보너스를 받게 된다. 현재와 이전 Dynasty의 총 예금 중 2/3 이상을 보유하고 있는 검증자가 새로운 블록에 대한 $Prepare$ tickets을 제안하면 해당 블록은 투표의 두 번째 단계로 들

어서게 된다. 주의해야할 점은 기본적으로 새 블록의 제안자가 새 블록에 대한 $Prepare$ tickets을 해당 새 블록에 제안해야 한다는 점이다.
- **두 번째 단계:** 모든 검증자가 새 블록에 대한 $Commit$ tickets(커밋 티켓)을 제안해야 한다. $Commit$ tickets을 제안한 후, 검증자는 1.5배의 보너스를 다시 받게 된다. 현재와 이전 Dynasty의 총 예금 중 2/3 이상을 보유하고 있는 검증자가 새 블록에 대한 $Commit$ tickets을 제안하면 해당 블록이 Finality(최종) 상태에 도달한다.

생태계 전체의 발전을 가속화하기 위해 블록 b $Prepare$ tickets 및 $Commit$ tickets과 블록 b 타임스탬프의 차이가 T를 초과하면 이 tickets은 만료된 것으로 간주되어 바로 무시된다.

5.3.4 포크 선택

PoD 알고리즘은 각 높이에서 블록 스코어에 따라 표준 체인을 선택한다. PoD 알고리즘은 항상 캐노니컬 체인(Canonical Chain)에 합류하기 위해 가장 높은 점수를 가진 블록을 선택하며, 높이 h에서 블록 b의 점수는 다음과 같다:

$$Score(b,h) = \sum_{(b',h')\in children(b)} Score(b',h') + \sum committed\ deposits\ in\ b \qquad (11)$$

즉, 해당 블록과 그 하위 블록 모두가 수신한 $Commit$ tickets(커밋 티켓)에 상응하는 예금의 합계(**그림.14** 참조)는 다음과 같다:

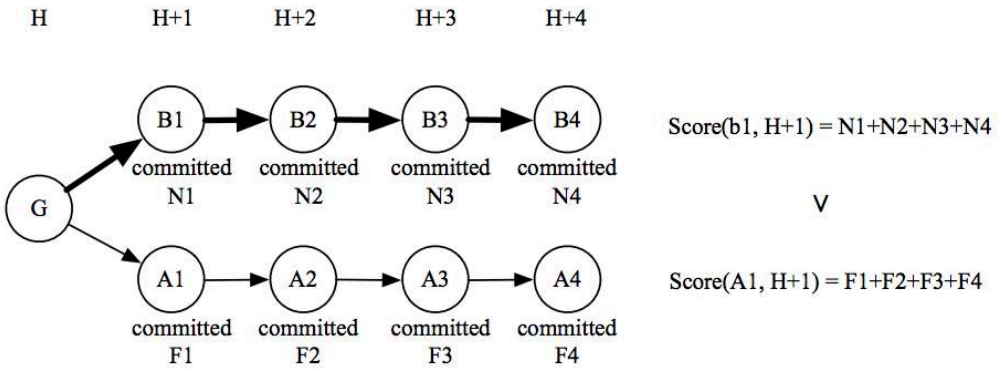


그림.14: 포크 선택 예제

5.3.5 투표 규칙

합의 실패 및 생태계 발전에 있어서 문제들을 초래할 수 있는 악의적인 손상을 피하기 위해, PoD는 캐스퍼(Casper)의 최소 삭감(Slashing) 조건을 참조하여 검증자의 합의 활동을 제한한다[35].

합의 과정에서 $Prepare$ 투표와 $Commit$ 투표가 다음의 구조로 이루어져 있다고 가정한다:

- $Prepare(H,v,vs)$: H는 현재 블록의 해시 값이고, v는 현재 블록의 높이를 나타내며, vs는 v의 특정 이전 블록의 높이를 나타낸다.
- $Commit(H,v)$: H는 현재 블록의 해시 값이고, v는 현재 블록의 높이를 나타낸다.

PoD 알고리즘은 전체 투표 과정에 대해 다음과 같은 4가지 기본 규칙을 규정한다:

- 단일 블록의 두 단계 합의 과정에는 엄격한 순서가 있다. 첫 단계에서 $Prepare(H,v,vs)$ 투표의 총 예금이 2/3에 달했을 경우에만, 검증자가 $Commit(H,v)$ 투표를 두 번째 단계로 전달할 수 있다.
- 여러 블록의 경우, 한 블록의 합의 과정이 완료될 때만 다음 블록의 합의 과정이 시작될 수 있다는 필수 규칙은 없다. 특정 순서에 따라 수행될 경우, 결합한 합의(interwoven consensus)가 제공 될 수 있다. 결합한 합의의 안정된 진행을 보장하기 위해 높이 vs에 대한 과정의 첫 번째 단계가 끝나고 $Prepare(H,v,vs)$ 투표의 비율이

2/3에 도달한 후에만 vs에 기반을 둔 $Prepare(H, v, vs)$가 이후 블록에 전달될 수 있다.
- 결합한 합의를 악용한 모든 노드의 악성 크로스 블록 투표를 피하기 위해, $Prepare(H, w, u)$ 투표를 u의 높이를 기준으로 전달 후, 모든 $Commit(H, v)$ 투표는 u에서 w까지의 범위 내의 높이로는 모든 블록에서 전달 될 수 없으므로 합의 과정의 높은 효율성과 질서를 보장한다.
- Nothing at Stake이라는 문제를 일으킬 수 있으므로 노드가 동시에 여러 지점에 하나의 예금으로 이체하는 것을 방지하기 위해 $Prepare(H1, v, vs1)$ 투표가 특정 높이에 도달한 후, 다른 $Prepare(H2, v, vs2)$ 투표를 다시 전달할 수 없다.

검증자가 위 규칙을 위반했다는 사실이 보고되고 위반 사실 여부가 확인이 된다면, 해당 검증자는 처벌을 받게 되고 모든 예금이 압수되며, 압수된 금액의 4%는 내부 신고자에게 보상으로 제공되고 나머지 금액은 처분된다.

5.4 PoD의 경제적 분석

5.4.1 인센티브 분석

PoD 알고리즘에 참여하는 검증자는 각각 해당하는 블록에서 NAS를 1배로 보상 받는다. $Prepare$ 단계를 완료하지 못하고 열악한 네트워크 트래픽이나 부정행위로 인해 $Commit$ 단계에 진입하는 경우 검증자는 0.5배의 NAS를 잃게 된다. 따라서 가치 노드가 되는 모든 검증자는 부정행위에 개입하지 않으면 네트워크 트래픽이 양호한 상태에서 많은 양의 수익을 확보하게 된다.

5.4.2 부정행위 분석

이중 지불(Double-spend) 공격

새로운 블록이 Finality(최종) 상태에 도달했을 때 가상상인(Merchant)이 해당 트랜잭션을 확인하고 인도할 경우, PoD 합의 알고리즘에 따라 이중 지불 공격을 완성 시키고 제로 비용의 구매 실현을 위해 부정행위로 지불해야 하는 최소 비용 다음과 같이 설명할 수 있다:

첫째, 부정행위를 하려면 Nebulas Rank의 랭킹을 Top N까지 올려야 하고 일정 금액의 NAS를 보증금으로 지불해야 하며, Dynasty D+2의 블록 유효성 확인에 참여하여 검증자가 되어야 한다.

그런 다음 부정행위를 하려면 의사 무작위(pseudo-random) 알고리즘에 의한 새로운 블록의 제안자로 선택되어야 한다. 이 시점에서 부정행위자는 동일한 높이를 가진 두 개의 새로운 블록을 제안해야 되며, 블록 하나는 hash1의 해시 값을 가지며 부정행위자에서 가상 상인으로 이어진 이전 트랜잭션을 포함하고 다른 블록은 hash2의 해시 값을 가지며 부정행위자에서 자신에게 트랜잭션을 전송한다.

마지막으로 hash1과 hash2 블록을 모두 Finality(최종) 상태로 만들기 위해 **그림.15**와 같이 해당 부정행위는 해당 Dynasty 전체 예금액 1/3을 검증자에게 뇌물로 주고 검증자가 $Commit$ 투표를 양 블럭에 전달하도록 한다.

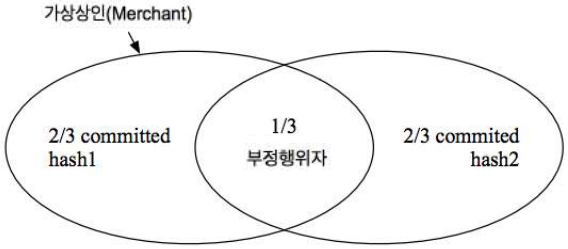


그림. 15: 이중 지불에 대한 패널티

51% 공격

PoW에서 51% 공격을 감행하려면 해시레이트(Hashrate)의 51%가 필요하다. PoS에서 51% 공격을 감행하려

면 전체 지분(예금)의 51%가 필요하다. 그러나 PoD에서 51% 공격을 감행하려면 검증자 무리 계정의 51%가 필요하다. 즉, 랭크 지수가 높은 충분한 수의 사용자가 Nebulas에서 상위 랭크에 오르고 각 계정마다 상위 랭크에 오르기 위한 생태계에 대한 기여도를 달성하거나 해당하는 만큼의 금액을 입금해야 한다. 즉, PoD에서 51%의 공격을 감행하는것은 굉장히 어려울 것이다.

단기적 공격(Short Range Attack)

PoD에서 각 높이의 블록은 특정 유효기간이 합의되어 있다. 따라서 PoD에서 장기적 공격을 수행하는 것은 거의 불가능하다. 하지만 유효기간 내에 단기적 공격을 감행할 수는 있다.

H+1 높이의 블록 유효기간 내에 단기적 공격자가 B 체인을 대체하기 위해 A 체인을 위조하려고 시도하면 공격자는 A1 블록 점수가 B1블록 점수보다 높도록 보장되어 있다. 다중 투표는 위중한 패널티를 받게 되므로 단기적 공격을 위해서는 공격자가 검증자에게 뇌물을 주는 행위가 있어야만 공격이 가능하다. PoD 합의 알고리즘의 안전성을 제시하기 위해 공격자가 다른 수의 블록을 되돌리는 데 드는 비용은 다음과 같이 분석된다:

공격자가 B1을 되돌릴 계획인 경우, **그림.16**에서 설명되었듯이 공격자가 지불해야 하는 최소 비용은 이중 지불 공격의 비용과 동일하다. 공격자가 H+1 높이 블록의 제안자가 되면, D0 Dynasty 검증자에게 1/3의 해당하는 뇌물을 주고 A1이 Finality(최종) 상태에 도달 할 수 있도록 검증자가 다중 투표를 수행하도록 해야 한다. 이 경우 최소 비용은 총 예금의 1/3이다.

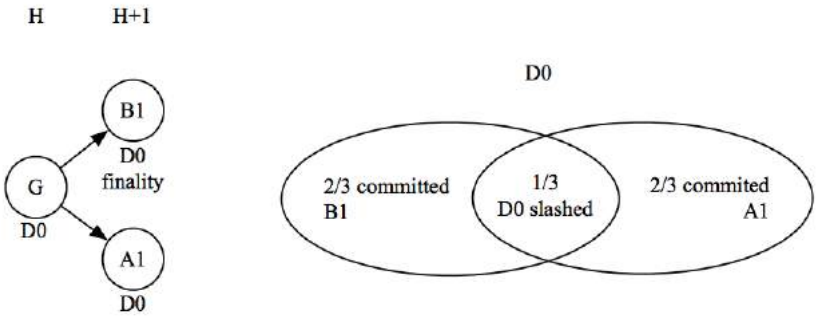


그림. 16: 단기적 공격을 통해 하나의 블록을 되돌리기

B1과 B2가 Finality(최종) 상태에 도달했으며 블록 내의 트랜잭션이 실행되었다고 가정해보자. 공격자가 B1-B2를 되돌리려면 다음 두 가지 상황을 고려해야 한다:

- 첫 번째 상황은 **그림.17(a)**에 나와 있다. 높이 H+1과 H+2가 같은 Epoch(세대)와 Dynasty에 존재할 시 공격자는 A1이 Finality 상태에 도달하도록 하기 위해 D0의 검증자에게 1/3의 해당하는 뇌물을 제공해야 한다. 한편, 이러한 1/3의 검증자는 패널티가 적용되며 예금은 완전히 몰수된다. A2의 유효기간 동안 예금의 합계는 A1 예금의 2/3와 같다. 현재 공격자가 B2와 동일한 금액의 투표를 확보하려는 경우와 부정행위를 하지 않는 경우에서는 나머지 모든 검증자에게 뇌물을 주고 총 예금의 2/3 이상을 잃어야 한다. 공격자가 이를 성공하더라도, A1 점수가 B1 점수보다 높도록 보장하는 것은 불가능하며, 공격자는 공격 실패에 대한 높은 리스크를 감수해야 한다.
- 두 번째 상황은 **그림.17(b)**에 나와 있다. 높이 H+1과 H+2가 서로 다른 Epoch와 Dynasty에 존재할 시, 공격자는 A1이 Finality 상태에 도달하도록 하기 위해 D0의 검증자에게 1/3의 해당하는 뇌물을 주어야 한다. 그 후, A2가 Finality 상태에 도달하도록 하기 위해 D1의 검증자에게 1/3의 해당하는 뇌물을 주어야 한다. 따라서 공격자는 이러한 공격을 성공시키기 위해 최소 총 예금의 2/3을 잃게 된다. 즉, 단기적 공격을 감행하여 Finality에 이른 블록 두 개를 무효화 하기 위해서 공격자는 총 예금의 2/3 이상을 지불해야 한다.

공격자가 B1-B3을 되돌리려는 경우(**그림.18**) 공격자는 A1이 Finality 상태에 도달하도록 D0의 검증자에게 1/3의 해당하는 뇌물을 주어야 하고, A2이 Finality 상태에 도달하도록 D1의 검증자에게 1/3의 해당하는 뇌물을 제공해야 한다. 마지막으로 공격자는 A3이 Finality 상태에 도달하도록 D1의 나머지 2/3의 검증자에게 뇌물을 제공해야 한다. 요약하자면 총 예금 금액의 4/3가 손실된다. 즉, 이러한 공격을 감행하는 것은 매우 어려울 것이다. 공격자가 가까스로 필요한 전파성을 만들어 내는데 성공한다고 해도 공격자는 A1의 점수가 B1의 점수보다 높게 책정될 수 있다고 보장할 수 없다. 따라서, 이러한 공격은 실패의 확률이 크다고 할 수 있다.

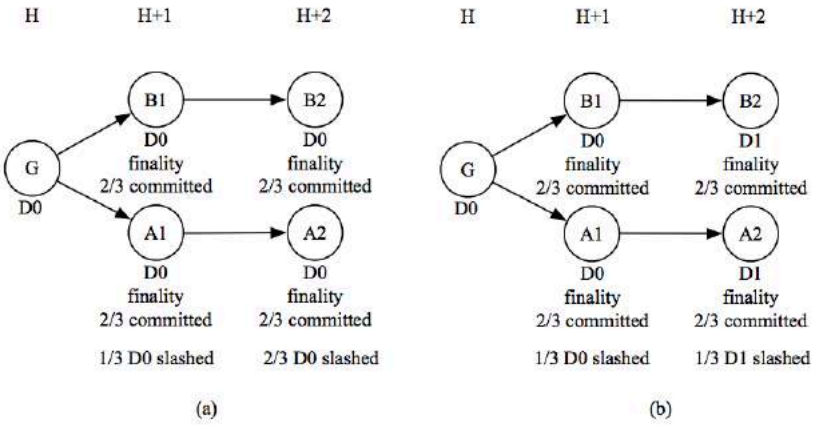


그림. 17: 단기적 공격을 통해 두 개의 블록을 되돌리기

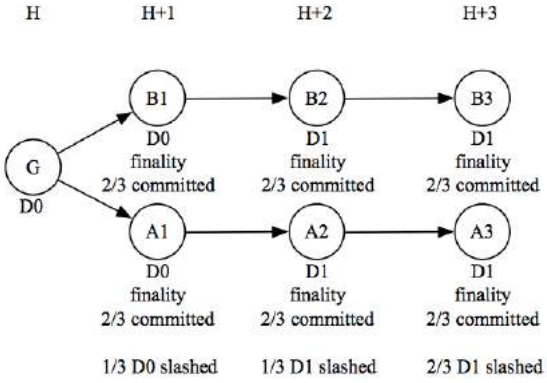


그림. 18: 단기적 공격을 통해 세 개의 블록을 되돌리기

공격자가 B1-BN을 되돌리려는 경우 블록 합의 유효기간 내에 제한되는 N은 보다 큰 수가 될 수 없다. N=3 일 때, 현재 Dynasty의 모든 검증자의 총 예금은 전적으로 몰수된다. 따라서 N>= 4 일 때 B1의 점수를 A1의 점수보다 높게 설정하고 B1-BN을 되돌리려면 공격을 완료 할 수 없으며, 이러한 공격을 감행하는 것은 의미 없는 일이라고 할 수 있다.

6 블록체인 검색엔진

6.1 소개

개발자가 스마트 컨트랙트를 배포하면, 스마트 컨트랙트에 대한 검색 요구가 급격히 상승하게 된다. 스마트 컨트랙트는 단순한 코드이며 기능적 설명은 포함하지 않으므로 검색 엔진 기술로 스마트 컨트랙트에 색인을 추가하는 일이 큰 어려움이 되고 있다. 스마트 컨트랙트를 적절히 색인화하기 위해 다음과 같은 방법을 사용한다:

- 스마트 컨트랙트와 관련된 웹 페이지 데이터를 크롤(Crawl)하여 데이터와 스마트 컨트랙트 간의 매핑을 설정한다.
- 개발자가 스마트 컨트랙트의 검증된 소스 코드를 업로드하며 코드의 기능과 의미를 분석하고, 소스 코드에 대한 색인을 만들어 유사한 컨트랙트에 대한 검색 기능을 제공하도록 권장한다. 소스 코드가 없는 스마트 컨트랙트의 경우 소스 코드용으로 디컴파일한다.
- 스마트 컨트랙트의 표준을 수립하면 사용자가 검색하여 해당 표준과 일치하는 컨트랙트를 찾을 수 있다. 또한 개발자가 스마트 컨트랙트 생성 중에 컨트랙트에 대한 정보를 제공하도록 독려한다.

```
contract SearchableContract {
    string public language;
    string public author;
    string public name;
    string public title;
    string public description;
    string public tags;
}
```

6.2 인프라(infrastructure)

현시점에서는 중앙화된 검색엔진이 최상의 사용자 경험을 제공하며 Nebulas Rank의 가치를 제시하는데 보다 적합하다고 생각한다. Nebulas 개발 팀은 검색 서비스를 개발하고, 모든 스마트 컨트랙트를 실시간으로 검색하며, 다국적 언어의 단어 분리를 수행하고, 전체 텍스트 색인을 작성하여 사용자 친화적인 웹 인터페이스를 제공하는데 전념하고 있다. Nebulas Rank 랭킹 알고리즘의 공정성과 각 노드의 검증 가능성으로 중앙 검색 서비스의 공정성을 보장하고, 검색 백엔드의 전체적인 코드는 커뮤니티에 모두 공개된다.

또한 제3자 개발자는 이를 기반으로 자체 검색 서비스를 만들 수 있다. **그림.19**는 검색 서비스의 아키텍처를 보여준다.

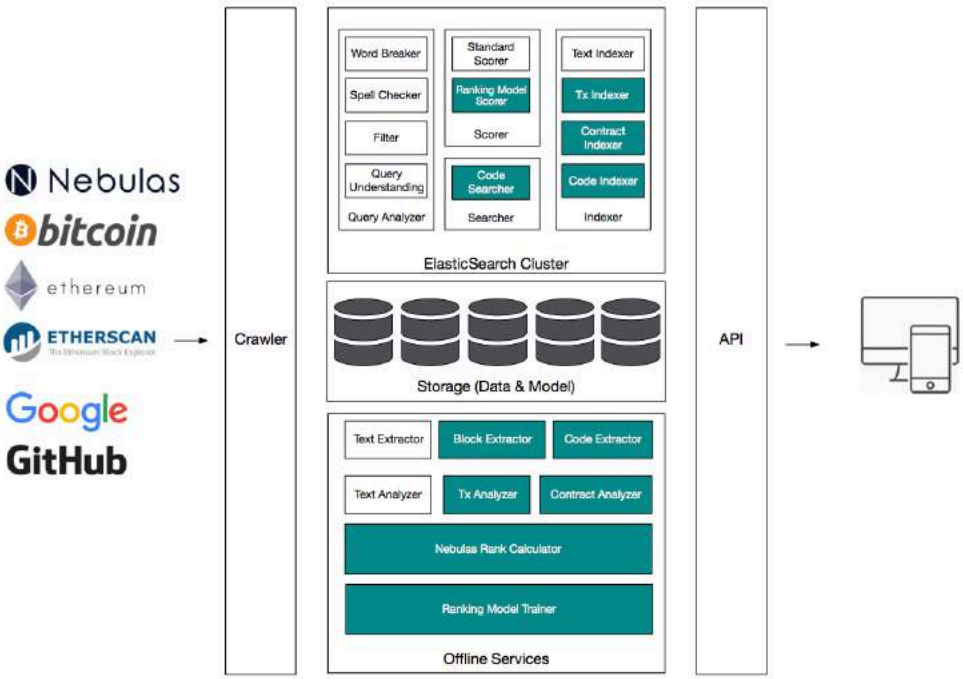


그림. 19: 검색 서비스의 아키텍쳐

- **크롤러(Crawler).** 블록체인 검색 엔진에서 크롤러(Crawler)의 데이터 소스는 두 가지 부류로 나눌 수 있다. 첫 번째는 블록 정보와 스마트 컨트랙트 코드를 블록체인에서 수집하는 것이다. 다른 한가지는 소개, Dapp 사용자 코멘트, 뉴스를 포함하여 공개 URL에서 스마트 컨트랙트에 대한 크롤링을 하는 것이다.
- **익스트랙터(Extractor).** 텍스트 추출기(Text Extractor), 블록 추출기(Block Extractor), 코드 추출기(Code Extractor)로 구성되어 있으며 각각 스마트 컨트랙트의 텍스트 정보, 블록 정보, 코드에 대한 추출 서비스를 제공한다.
- **분석기(Analyzer).** 텍스트 분석기(Text Analyzer), Tx 분석기(Tx Analyzer), 컨트랙트 분석기(Contract Analyzer)로 구성되며 각각 텍스트 정보, 블록 트랜잭션 정보, 스마트 컨트랙트 분석의 기능을 제공한다. 스마트 컨트랙트 분석기의 경우 컨트랙트 디컴파일, 소스 코드 추출, 의미 분석 등의 기능을 제공한다.
- **Nebulas Rank 계산기(Nebulas Rank Calculator).** Nebulas Rank 계산기 서비스를 지칭한다. 이 서비스는 각각의 오프라인 비컨트랙트 계정 및 컨트랙트 계정의 Nebulas Rank를 계산하는데 사용된다.
- **랭킹 모델 트레이너(Ranking Model Trainer).** 랭킹 모델 트레이너 서비스를 의미한다. 랭킹 규칙은 매칭 필드(matching field), 텍스트 관련성, 컨트랙트 순위의 Nebulas Rank 랭크 가치, 컨트랙트의 트랜잭션 양, 빈도 및 심도, 컨트랙트를 통해 트랜잭션을 수행하는 사용자의 Nebulas Rank 랭크, 컨트랙트 보안과 같은 여러 요소를 반영한다. 사용자의 실제 사용 조건에 따라 머신 러닝 알고리즘(GBDT 및 인공 신경망(ANN)은 선택 사항)은 랭킹 및 점수 모델을 훈련에 사용되며 사용자 피드백에 따라 지속적으로 향상된다. 훈련된 모델은 검색 서비스의 스코러(Scorer)에 사용된다.
- **쿼리 분석기(Query Analyzer).** 다국적 언어 단어 분리기(Word Breaker) 및 맞춤법 검사기(Spell Checker)를 포함하는 키워드 분석 서비스를 의미한다.
- **인덱서(Indexer).** 분석기에서 적절한 인덱스를 만들고 전체 및 증분 인덱싱을 모두 지원한다.
- **스코러(Scorer).** 두 가지 레벨로 분류된다. 레벨1 표준 스코러(Level-1 Standard Scorer)는 엘라스틱서치(ElasticSearch) 클러스터에서 효율적이고 효과적인 순위를 부여해 최대한 많은 후보 결과를 불러 오기 위해 엘라스틱서치에서 후보 결과 집합을 소환한다. 레벨-1은 수천 개의 결과를 불러올 수 있다. 레벨-2 랭킹 모델 스코러(Ranking Model Scorer)는 오프라인 랭크 모델을 사용하여 각 레벨1 후보 결과 집합의 순위를 계산하고 순서를 변경한다. 이 수준에서 계산된 결과는 조사 정확도(sounding accuracy)를 특징으로 하며 사용자가 직접 계산된 결과를 사용할 수 있다.
- **조사자(Searcher).** 엘라스틱서치 클러스터와 통신을 담당하고 검색 결과를 검색 프론트 엔드에 패킹 및 반환한다.
- **API** 외부 응용 프로그램에 포괄적 검색 API 서비스를 제공한다.
- **엘라스틱서치 클러스터(ElasticSearch Cluster).** ES는 서버 클러스터를 의미한다. Nebulas 개발 팀은 오픈 소스 검색 엔진 엘라스틱서치를 사용하여 전체 텍스트 인덱싱을 지원할 계획이다.

6.3 Nebulas 트렌드

Nebulas는 Nebulas Rank를 결합하여 트렌드 목록을 생성하고 사용자에게 시각적인 다차원 값을 블록체인에 제공한다.

- **비컨트랙트 계정 Nebulas Rank 목록**. 일별 Nebulas Rank 목록과 Nebulas Rank 급상승 및 급하강 목록을 표시한다. 또한 이 순위 목록은 각 컨트랙트의 순위 변동 경향과 전체 네트워크의 상태 변화 경향을 시각화하여 데이터를 제공한다.
- **컨트랙트 계정 Nebulas Rank 순위 목록**. 비컨트랙트 계좌의 Nebulas Rank 값을 기반으로 Nebulas Rank 목록은 전체 네트워크에서 컨트랙트 계좌의 Nebulas Rank 목록, 급상승 및 급하락 목록, 각 컨트랙트의 변동 경향 및 스마트 컨트랙트의 수량 및 사용 빈도에 대한 경향 차트를 계산한다. 또한 토큰 컨트랙트 목록 및 시장 컨트랙트 추정 목록과 같은 기타 스마트 컨트랙트 목록을 제시하여 보다 넓은 범위의 정보를 표시한다.
- **스마트 컨트랙트 개발자 목록**. 컨트랙트 계정 목록에 따라, 스마트 컨트랙트 개발자 목록은 컨트랙트 개발자의 기여 목록과 급상승 목록의 기여를 계산하여 보다 뛰어난 컨트랙트 개발자와 Dapp을 표시한다.

6.4 키워드 검색어

키워드를 제공하거나 제목, 저자 또는 기능과 같은 스마트 컨트랙트에 대한 텍스트 정보를 설명하여 사용자는 대규모 스마트 컨트랙트에서 일치하는 컨트랙트를 찾을 수 있다. 현재 성숙하고 정교한 알고리즘 및 기술을 텍스트 검색에 사용할 수 있다. 자연어 처리 및 역색인 기술을 사용하여 대규모 스마트 컨트랙트의 인덱스 데이터베이스에서 효율적으로 검색하고 정렬 할 수 있다. 여기에는 다음 핵심 기술이 포함된다:

1. 주제 지향의 분산 크롤러 기술
2. 다국적 언어 단어 분리 기술: 단어 분리는 영문 단어에는 비교적 간단하다. 중문 단어의 단어 분리의 경우, 양의 최대 일치(positive maximum matching), 음의 최대 일치(negative maximum matching), 최단 경로 단어 분리 및 통계 단어 분리와 같은 알고리즘을 사용할 수 있다.
3. 검색 용어 수정 및 의미 이해
4. 역색인 및 분산 검색 아키텍처
5. 랭킹 알고리즘을 사용한 검색 결과 정렬

이러한 기술 중 랭킹 알고리즘은 Nebulas Rank와 함께 설계된다. 특히 Nebulas는 블록체인에서 블록체인 트랜잭션 그래프를 만들기 위해 내부 사용자(intra-user) 전송을 사용한다. 이것은 인터넷 세계에서 웹 페이지 참조 관계와 유사한 것이다. 그런 다음 2.3 섹션에서 설명한 Nebulas Rank 랭킹 알고리즘을 활용하여 비컨트랙트 랭크의 Nebulas Rank 랭크을 계산하고 4.2 섹션에서 설명된 컨트랙트 랭킹 알고리즘을 사용하여 컨트랙트 순위를 계산한 다음 마지막으로 계산 결과를 검색 결과 정렬에 사용한다.

6.5 유사 스마트 컨트랙트 검색

개발자와 특정 사용자의 경우 컨트랙트의 코드 조각에 따라 비슷한 기능을 가진 스마트 컨트랙트를 검색하고자 할 수 있다. 일반적인 키워드 검색과 달리 코드 유사성은 그 특성을 가지고 있다. 유사한 스마트 컨트랙트에 대한 검색 기능을 구현하려면 특정 알고리즘을 사용하여 숫자 또는 백분율로 코드 유사성을 측정해야 한다.

오늘날의 학계에서 코드 유사성 알고리즘은 주로 문자열 편집 거리(string edit distance), 토큰 시퀀스 유사성, 추상 구문 트리 유사성, 프로그램 종속성 그래프 유사성으로 분류된다. 이 알고리즘은 서로 다른 차원의 코드 텍스트, 구조 및 구문에 대한 유사성을 설명한다. 이 네 가지 주요 알고리즘을 결합하여 스켈레톤 트리(Skeleton Tree), 타입 서명(Type Signature), 라이브러리 콜(Library Calls)과 같은 Nebulas 컨트랙트의 코드 유사성 12가지 특징을 제안한다. 자세한 내용은 부록 B를 참조하십시오.

키워드의 검색 결과와 마찬가지로 스마트 컨트랙트의 검색 결과도 동일한 컨트랙트 순위 알고리즘을 사용하여 최종 결과를 정렬한다.

7 기본 서비스 및 개발자 툴

7.1 도메인 네임 서비스

블록체인의 익명성으로 인해 컨트랙트 주소는 길고 의미 없는 문자열이기 때문에 사용하기 쉽지 않다. 이러한 이유로 사용자는 의도하지 않은 자산 입출금이나 잘못된 대상과의 상호 작용으로 인해 금전적 손실을 입는 등의 오작동 발생에 취약하다. 즉, 기억하기 쉬운 도메인 네임을 사용하면 사용자는 보다 나은 경험을 얻을 수 있다. Nebulas 개발 팀은 스마트 컨트랙트를 사용하여 자유와 개방성을 보장하면서도 이용하기 편리한 Nebulas 네임 서비스(Nebulas Name Service)(NNS)라는 DNS와 유사한 도메인 시스템을 체인에 구현할 것이다. 제3자 개발자는 NNS를 기반으로 또는 자체적으로 도메인 네임 분석 서비스(domain name resolution services)를 구현할 수 있다.

예를 들어, 엘리스(Alice)는 자신의 계정 주소 0xdf4d22611412132d3e9bd322f82e2940674ec1bc03b20e40에 대하여 “alice.nns”라는 도메인 네임을 신청할 수 있다. 밥(Bob)이 엘리스(Alice)에게 돈을 송금하려면 수취인 정보에 "alice.nns"를 간단하게 입력하기만 하면 NNS 서비스를 통해 정확한 수취인 주소로 금액을 이체할 수 있다.

NNS 어플리케이션 규칙은 다음과 같다:

- 최상위 서브 도메인 네임은 제한될 것이다. *.nns, *.com, * .org, * .edu와 같은 도메인의 신청은 불가능 하다. 따라서 사용자는 두 번째 수준의 서브 도메인 네임만 신청할 수 있다.
- NNS 서비스가 활성화되면 사용자는 가용성을 위해 도메인 네임을 쿼리 처리 할 수 있다. 빈 도메인 네임의 경우 사용자는 스마트 컨트랙트를 통해 입찰 할 수 있다. 모든 사용자가 다른 사용자의 입찰가를 쿼리 처리하고 언제든지 자신의 입찰가를 업데이트 할 수 있도록 입찰 프로세스가 열려 있다.
- 입찰 기간이 만료되면 가장 높은 입찰자가 도메인 네임을 얻고 스마트 컨트랙트는 사용자의 입찰 예금을 잠그게 된다. 도메인 네임의 유효 기간은 1년이다. 1년 후 사용자는 갱신 여부를 자유롭게 결정할 수 있다. 갱신하길 원하는 경우, 유효 기간이 1년 더 연장된다. 그렇지 않은 경우 입찰 예금은 사용자 계정으로 환급되며 도메인 네임은 다시 사용할 수 있게 공개된다.
- 사용자는 언제든지 도메인 네임에 대한 소유권을 포기할 수 있다. 이 경우 입찰 예금은 사용자 계정으로 자동 환불되며 도메인 데이터가 삭제 된 후에 도메인 네임이 다시 사용 가능하도록 해제된다.
- 사용자는 보상 여부와 상관없이 도메인 네임의 소유권을 양도 할 수 있다. Nebulas는 도메인 네임 거래에 개입하지 않는다.

7.2 라이트닝 네트워크

현재 모든 퍼블릭 블록체인 네트워크는 시스템 확장성 문제에 직면 하고 있다. 예를 들어 비트코인 네트워크는 초당 7개의 트랜잭션만 처리 할 수 있으며 이더리움은 초당 15개의 트랜잭션만 처리 할 수 있다. PoS 유사 합의 알고리즘을 도입함으로써 채굴 계산을 피할 수 있고, 합의 속도가 크게 향상 될 수 있다.

그러나 퍼블릭 블록체인은 실제 세계에서 대량의 소액 결제 시나리오면에 있어서는 여전히 큰 과제를 안고 있다. 2015년 2월에 출시된 라이트닝 네트워크[47]는 트랜잭션 당사자 간의 소액 결제 채널 네트워크를 구축하여 트랜잭션 당사자 간의 대량 지불이 네팅 방법에서 직접적으로, 반복적으로, 빈번히, 양방향으로 블록체인 밖(off the blockchain)에서 확인될 수 있다. 트랜잭션이 성사되어야 할 때 최종 결과는 확인을 위해 블록체인에 제출된다. 이론적으로 이러한 프로세스는 초당 수백만의 전송을 담당할 수 있다. 양 당사자간에 점대점(point-to-point) 지불 채널이 없는 경우, 양 당사자를 연결하고 여러 지불 채널로 구성된 지불 경로를 사용하여 당사자간에 안정적인 자금 이체를 수행 할 수 있다. 라이트닝 네트워크는 비트코인과 이더리움에서 PoC 단계를 거쳤다.

Nebulas는 라이트닝 네트워크를 블록체인의 인프라로 구현하고 유연한 설계를 제공한다. 제3자 개발자는 라이트닝 네트워크 기본 서비스를 사용하여 Nebulas에서 트랜잭션 빈도가 높은 시나리오용 어플리케이션을 개발할 수 있다. 또한 Nebulas는 라이트닝 네트워크를 지원하는 세계 최초의 지갑 앱(wallet app)을 출시 할 예정이다.

7.3 개발자 툴

완전한 개발자 툴 세트는 블록체인 앱 개발자에게 필수적이다. 지금까지, 퍼블릭 블록체인용 개발자 툴 체인은 불완전하여 대부분의 개발자에게 큰 어려움을 주고 있다. Nebulas 개발 팀은 스마트 컨트랙트용 독립 개발 IDE, 브

라우저 차단, 다양하고 인기 있는 IDE(Eclipse, JetBrains, Visual Studio, Sublime Text, VIM 및 Atom 포함) 용 플러그인, 디버그 툴, 시뮬레이터, 스마트 컨트랙트용 공식 인증 툴, 다양한 고급언어용 백그라운드 SDK, 모바일 엔드용 SDK 등 풍부한 개발자 툴을 제공할 예정이다.

8 Nebulas(NAS) 토큰

Nebulas 네트워크에는 자체 내장 토큰인 NAS가 있다. NAS는 네트워크에서 두 가지 역할을 수행한다. 첫째, NAS는 네트워크의 오리지널 머니(original money)로 사용자들에게 자산 유동성을 제공하고 PoD 북키핑 관리자 및 Developer Incentive Protocol의 인센티브 토큰으로서의 기능을 한다. 둘째, NAS는 스마트 컨트랙트 체결에 대한 계산 수수료로 청구된다. NAS의 최소 단위는 10^{-18} NAS이다.

원래 NAS는 이더리움 플랫폼에서 ERC20 토큰으로 판매되었으며 최대 NAS 집계는 $X = 10^8$. 이다. 할당 모드는 다음과 같다:

1. **커뮤니티 건설**: Nebulas 스폰서 팀의 지시에 따라 80%X 토큰이 Nebulas 커뮤니티 블록체인 분산 어플리케이션 (DApp)의 생태계 인큐베이션 및 인센티브, 개발자 공동체 건설, 비즈니스 및 산업 협력, 마케팅 및 프로모션 활동, 학술 연구, 교육 투자, 법률 및 규정, 커뮤니티 및 조직에 대한 투자 등의 Nebulas 생태계 구축에 사용된다. 특히, 커뮤니티상의 영향력이 있는 투자자에게는 5%X가 판매되며, Nebulas 커뮤니티 개발 기금으로 5%X가 판매되고, 70%X는 보류해 둘 것이다..
2. 스폰서 및 개발 팀 인센티브: Nebulas 개발을 통해 스폰서 및 개발 팀은 프로젝트 조직 구조, 기술 연구 및 개발 및 생태 운영 측면에서 지속적으로 인적 기여 및 물질적 자원 형태의 기여를 할 것이다. 토큰 할당에 대해서는 팀 인센티브 목적으로 20%X가 보류되어 있다. NAS의 해당 부분은 초기에 잠겨져 있으며 커뮤니티에 처음으로 NAS를 판매하고 1년 후에 잠금이 해제되며 3년 후 점진적으로 스폰서 및 개발 팀에 배포된다.

Nebulas 네트워크 공식 출시 후, 이더리움 ERC20 NAS 토큰이 있는 사용자는 관련 자격 증명을 통해 Nebulas 네트워크에서 동등한 수량의 NAS를 청구 할 수 있으며 이더리움 ERC20 NAS 토큰은 회수될 것이다. Nebulas의 네트워크가 발전함에 따라 NAS는 다음과 같은 방식으로 배분될 것이다.

1. Proof of Devotion(PoD) 인센티브: 북키퍼를 위한 연간 3%X의 NAS 인센티브
2. Developer Incentive Protocol(DIP) 인센티브: 스마트 컨트랙트 개발자에게 매년 1%X NAS 인센티브

9 결론

Nebulas의 자산

매우 추상적인 관점에서 블록체인은 **분산 방식으로 데이터의 확인을 정확히 하는 것**이며, 토큰은 **데이터 확인의 "가치"를 운반하는 역할**을 한다. 인터넷은 데이터 통신을 해결하고 블록체인은 데이터 확인이라는 문제를 해결한다. 블록체인은 처음으로 퍼블릭 데이터를 프라이빗 데이터로 변환한다. 이러한 프라이빗 데이터는 구글(Google), 아마존 (Amazon) 및 페이스북(Facebook)과 같은 대기업에서 더 이상 임의로 분석되지 않고 활용되지 않는다.

퍼블릭 블록체인으로 대표되는 블록체인의 핵심은 **"커뮤니티+토큰+응용프로그램"**이다. 커뮤니티는 본질적으로 개방형, 오픈 소스, 공유 및 비영리 단체라는 개념을 고수한다. 이러한 특성은 기존의 모든 비즈니스 생태계와 완전히 다르다. 앞서 언급했듯이, 토큰은 데이터 확인의 "가치"를 운반하는 역할을 한다. 가상 화폐와 전자 화폐의 부속물로만 사용되는 것을 넘어서서 미래에는 다양한 시나리오가 마련될 것이다. 어플리케이션은 단순히 블록체인 어플리케이션 시나리오의 기술적 구현을 나타낸다. 앞서 언급한 두 가지 요소가 결합되지 않는다면 어플리케이션만으로는 **블록체인 시스템의 매력을 충분히 누릴 수 없다.**

퍼블릭 블록체인으로 대표되는 블록체인 시스템은 블록체인의 미래이다. "무신뢰(trustless)"및 "무허가(Permissionless)"의 특성은 블록체인 시스템의 실제 가치이다. 반대로 대부분의 컨소시엄 블록체인/엔터프라이즈 블록체인은 "신뢰 기반"및 "허가 기반"이므로 기존 패턴을 벗어날 수 없으며 그저 다소 향상된 혁신으로 간주된다. 퍼블릭 블록체인 시스템은 기존 협력 관계를 뒤엎고 블록체인의 최대 가치를 반영하는 "파괴적인" 혁신으로 간주된다.

Nebulas의 약속

전 세계적으로 최초의 블록체인 검색 엔진로서 Nebulas는 블록체인 가치의 숨겨진 차원을 탐구하고 가치 기반 블록체인 운영 체제, 검색 엔진 및 기타 관련 광범위한 어플리케이션을 구축하기 위해 최선을 다하고 있다.

이 약속을 지키기 위해 Nebulas Rank를 설정하여 블록체인 세계의 가치 측정을 설정하고, Nebulas Force를 통해 블록체인의 자체적 진화 능력을 강화하고, Developer Incentive Protocol 및 Proof of Devotion(PoD)를 개발하여 블록체인의 가치 향상을 유도하고, Nebulas 검색 엔진을 구성하여 사용자가 다른 차원의 블록체인 가치를 탐구하는데 도움을 줄 것이다.

Nebulas의 신념

현재 과학과 기술의 진화는 우리로 하여금 보다 높은 수준의 자유와 평등이 존재하는 삶으로 인도할 것이다. 그런 삶을 위해 우리를 인도할 주요 기술 중 하나인 블록체인은 점진적으로 블록체인만의 비교불가한 장점을 발휘하게 될 것이다. 이러한 기술적 진화의 과정에 참여할 기회를 누릴 수 있다는 것은 Nebulas에게 가장 큰 기회이자 모티브이다.

인터넷과 마찬가지로 블록체인은 폭발적인 사용자/앱 단계로 접어들 수 있다. 블록체인 기술은 차세대 "스마트 네트워크"의 기본 프로토콜이 될 것이며, 향후 5년, 10년 사이에 사용자가 10억 명에 도달할 것이다. 따라서, 중대 변화를 일으킬 기회와 도전은 향후 5년 내에 지속적으로 나타날 것이다.

블록체인은 살아있는 유기체이며 항상 진화하고 움직이는 경제체이다. 블록체인이 무엇을 할 수 있는지 묻고 대신 블록체인을 위해 무엇을 할 수 있는지 탐구해보자. Nebulas는 블록체인의 기술적 탐구에 있어 이 모든 것을 여러분과 함께 공유할 수 있게 되어 대단히 기쁘게 생각한다.

참고문헌

[1] Luke Anderson *et al.* "New kids on the block: an analysis of modern blockchains." In: (2016). arXiv: 1606.06530. URL: http://arxiv.org/abs/1606.06530.

[2] Annika Baumann, Benjamin Fabian, and Matthias Lischke. "Exploring the Bitcoin network." In: *WEBIST 2014 - Proceedings of the 10th International Conference on Web Information Systems and Technologies* 1 (2014), pp. 369–374. ISSN: 9789897580239 (ISBN). DOI: 10.5220/0004937303690374. URL: https://www.engineeringvillage.com/blog/document.url?mid=cpx%7B%5C_%7D9ce5505146fd48dcbdM557010178163125%7B%5C&%7Ddatabase=cpx.

[3] Morten L Bech and Enghin Atalay. "The topology of the Federal Funds markets." In: *Economic Policy Review* 14.2 (2008).

[4] Phillip Bonacich. "Factoring and weighting approaches to status scores and clique identification." In: *Journal of Mathematical Sociology* 2.1 (1972), pp. 113–120.

[5] Stephen P. Borgatti. "Centrality and network flow." In: *Social Networks* 27.1 (2005), pp. 55–71. ISSN: 03788733. DOI: 10.1016/j.socnet.2004.11.008.

[6] Stephen P. Borgatti and Martin G. Everett. "A Graph-theoretic perspective on centrality." In: *Social Networks* 28.4 (2006), pp. 466–484. ISSN: 03788733. DOI: 10.1016/j.socnet.2005.11.005.

[7] Michael Boss, Martin Summer, and Stefan Thurner. "Contagion Flow Through Banking Networks." In: *Lecture Notes in Computer Science 3038* (2004), pp. 1070–1077. ISSN: 03029743. DOI: 10.1016/j.jfi.2008.06.003. arXiv: 0403167v1 [arXiv:cond-mat].

[8] Michael Boss *et al.* "The Network Topology of the Interbank Market." In: *Quantitative Finance* 4.6 (2004), pp. 677–684. ISSN: 1469-7688. DOI: 10.1080/14697680400020325. arXiv: 0309582 [cond-mat]. URL: http://arxiv.org/abs/cond-mat/0309582.

[9] Sergey Brin and Lawrence Page. "Reprint of: The anatomy of a large-scale hypertextual web search engine." In: *Computer Networks* 56.18 (2012), pp. 3825–3833. ISSN: 13891286. DOI: 10.1016/j.comnet.2012.10.007. arXiv: 1111.6189v1.

[10] Vitalik Buterin. "Ethereum: A next-generation smart contract and decentralized application platform." In: *URL https://github. com/ethereum/wiki/wiki/% 5BEnglish% 5D-White-Paper* (2014).

[11] Vitalik Buterin *et al. Ethereum white paper*. 2013.

[12] Lijun Chang *et al.* "pSCAN: Fast and Exact Structural Graph Clustering." In: *IEEE Transactions on Knowledge and Data Engineering* 29.2 (2017), pp. 387–401.

[13] Tao Hung Chang and Davor Svetinovic. "Data Analysis of Digital Currency Networks: Namecoin Case Study." In: *Proceedings of the IEEE International Conference on Engineering of Complex Computer Systems, ICECCS* (2017), pp. 122–125. DOI: 10.1109/ICECCS.2016.023.

[14] Duan Bing Chen *et al.* "Identifying influential nodes in large-scale directed networks: The role of clustering." In: *PLoS ONE* 8.10 (2013), pp. 1–10. ISSN: 19326203. DOI: 10.1371/journal.pone.0077455.

[15] Michel Chilowicz, Etienne Duris, and Gilles Roussel. "Syntax tree fingerprinting for source code similarity detection." In: *Program Comprehension, 2009. ICPC'09. IEEE 17th International Conference on*. IEEE. 2009, pp. 243–247.

[16] *Etherscan - The Ethereum Block Explorer*. https://etherscan.io/. Accessed: 2017-08-01.

[17] Giorgio Fagiolo. "The International-Trade Network: Gravity Equations and Topological Properties." In: (2009). arXiv: 0908.2086. URL: http://arxiv.org/abs/0908.2086.

[18] Jeanne Ferrante, Karl J Ottenstein, and Joe D Warren. "The program dependence graph and its use in optimization." In: *ACM Transactions on Programming Languages and Systems (TOPLAS)* 9.3 (1987), pp. 319–349.

[19] Danno Ferrin. "A Preliminary Field Guide for Bitcoin Transaction Patterns." In: *Texas Bitcoin Conference* (2015). URL: http://texasbitcoinconference.com.

[20] Michael Fleder, Michael S. Kester, and Sudeep Pillai. "Bitcoin Transaction Graph Analysis." In: *... -Transaction-Graph-Analysis. Pdf"...* (2015), pp. 1–8. arXiv: 1502.01657. URL: http://arxiv.org/abs/1502.0165%7B%5C%%7D5Cnhttp://people.csail.mit.edu/spillai/data/papers/bitcoin-project-paper.pdf%7B%5C%%7D5Cnhttp://arxiv.org/abs/1502.00165%7B%5C%%7D5Cnhttp://arxiv.org/abs/1502.01657.

[21] L Freeman. "A set of measures of centrality: I. Conceptual clarification." In: *Soc. Networks* 1 (1979), pp. 215–239.

[22] Linton C Freeman. "A set of measures of centrality based on betweenness." In: *Sociometry* (1977), pp. 35–41.

[23] Linton C Freeman. "Centrality in social networks conceptual clarification." In: *Social networks* 1.3 (1978), pp. 215–239.

[24] Linton C Freeman, Stephen P Borgatti, and Douglas R White. "Centrality in valued graphs: A measure of betweenness based on network flow." In: *Social networks* 13.2 (1991), pp. 141–154.

[25] Sudipto Guha *et al.* "Approximate XML joins." In: *Proceedings of the 2002 ACM SIGMOD international conference on Management of data*. ACM. 2002, pp. 287–298.

[26] Bernhard Haslhofer, Roman Karl, and Erwin Filtz. "O Bitcoin Where Art Thou? Insight into Large-Scale Transaction Graphs." In: ().

[27] Leo Katz. "A new status index derived from sociometric analysis." In: *Psychometrika* 18.1 (1953), pp. 39–43.

[28] S King and S Nadal. "Peercoin–Secure & Sustainable Cryptocoin." In: *Aug-2012 [Online]. Available: https://peercoin.net/whitepaper* ().

[29] Jon M Kleinberg. "Authoritative sources in a hyperlinked environment." In: *Journal of the ACM (JACM)* 46.5 (1999), pp. 604–632.

[30] Lothar Krempel. "Exploring the Dynamics of International Trade by Combining the." In: December (2002), pp. 1–22.

[31] Qian Li *et al.* "Identifying influential spreaders by weighted LeaderRank." In: *Physica A: Statistical Mechanics and its Applications* 404 (2014), pp. 47–55. ISSN: 03784371. DOI: 10.1016/j.physa.2014.02.041. arXiv: arXiv:1306.5042v1.

[32] *llvm*. https://llvm.org/. Accessed: 2017-08-01.

[33] Linyuan Lü *et al.* "Vital nodes identification in complex networks." In: *Physics Reports* 650 (2016), pp. 1–63. ISSN: 03701573. DOI: 10.1016/j.physrep.2016.06.007. arXiv: 1607.01134.

[34] Sarah Meiklejohn *et al.* "A fistful of Bitcoins: Characterizing payments among men with no names." In: *Proceedings of the Internet Measurement Conference - IMC '13* 6 (2013), pp. 127–140. ISSN: 15577317. DOI: 10.1145/2504730.2504747. URL: http://dl.acm.org/citation.cfm?id=2504730.2504747.

[35] *Minimal Slashing Conditions*. https://medium.com/@VitalikButerin/minimal-slashing-conditions-20f0b500fc6c. Accessed: 2017-08-01.

[36] L Morten, J Robert, and E Walter. "The topology of interbank payment flows." In: (2006).

[37] Satoshi Nakamoto. "Bitcoin: A Peer-to-Peer Electronic Cash System." In: *Www.Bitcoin.Org* (2008), p. 9. ISSN: 09254560. DOI: 10.1007/s10838-008-9062-0. arXiv: 43543534534v343453. URL: https://bitcoin.org/bitcoin.pdf.

[38] *NEM Technical Reference*. http://nem.io/NEM_techRef.pdf. Accessed: 2017-08-01.

[39] Mark Newman. *Networks: an introduction*. Oxford university press, 2010.

[40] Mark EJ Newman. "A measure of betweenness centrality based on random walks." In: *Social networks* 27.1 (2005), pp. 39–54.

[41] Dá Niel Kondor *et al.* "Do the Rich Get Richer? An Empirical Analysis of the Bitcoin Transaction Network." In: *PLoS ONE* 9.2 (2014). DOI: 10.1371/journal.pone.0086197.

[42] Athanasios N. Nikolakopoulos and John D. Garofalakis. "NCDawareRank." In: *Proceedings of the sixth ACM international conference on Web search and data mining - WSDM '13* February 2013 (2013), p. 143. DOI: 10.1145/2433396.2433415. URL: http://dl.acm.org/citation.cfm?doid=2433396.2433415.

[43] Jae Dong Noh and Heiko Rieger. "Random walks on complex networks." In: *Physical review letters* 92.11 (2004), p. 118701.

[44] Micha Ober, Stefan Katzenbeisser, and Kay Hamacher. "Structure and Anonymity of the Bitcoin Transaction Graph." In: *Future Internet* 5.2 (2013), pp. 237–250. ISSN: 1999-5903. DOI: 10.3390/fi5020237. URL: http://www.mdpi.com/1999-5903/5/2/237/.

[45] Lawrence Page *et al. The PageRank citation ranking: Bringing order to the web*. Tech. rep. Stanford InfoLab, 1999.

[46] Thai Pham and Steven Lee. "Anomaly detection in bitcoin network using unsupervised learning methods." In: *arXiv preprint arXiv:1611.03941* (2016).

[47] Joseph Poon and Thaddeus Dryja. "The bitcoin lightning network: Scalable off-chain instant payments." In: *Technical Report (draft)* (2015).

[48] Marc Pröpper, Iman van Lelyveld, and Ronald Heijmans. "Towards a network description of interbank payment flows." In: (2008).

[49] Dorit Ron and Adi Shamir. "Quantitative Analysis of the Full Bitcoin Transaction Graph." In: ().

[50] Gert Sabidussi. "The centrality index of a graph." In: *Psychometrika* 31.4 (1966), pp. 581–603.

[51] Computer Science and The Technion. "The Stochastic Approach for Link-Structure Analysis (SALSA) and the TKC E ect." In: (2001).

[52] M. Ángeles Serrano, Marián Boguñá, and Alessandro Vespignani. "Patterns of dominant flows in the world trade web." In: *Journal of Economic Interaction and Coordination* 2.2 (2007), pp. 111–124. ISSN: 1860711X. DOI: 10.1007/s11403-007-0026-y. arXiv: 0704.1225.

[53] Ma Angeles Serrano and Marián Boguñá. "Topology of the world trade web." In: *Physical review. E, Statistical, nonlinear, and soft matter physics* 68.1 Pt 2 (2003), p. 015101. ISSN: 1063-651X. DOI: 10.1103/PhysRevE.68.015101. arXiv: 0301015 [cond-mat].

[54] Hiroaki Shiokawa, Yasuhiro Fujiwara, and Makoto Onizuka. "SCAN++: efficient algorithm for finding clusters, hubs and outliers on large-scale graphs." In: *Proceedings of the VLDB Endowment* 8.11 (2015), pp. 1178–1189.

[55] *The Stage 1 Casper Contract*. https://github.com/ethereum/casper/. Accessed: 2017-08-01.

[56] Florian Tschorsch and Björn Scheuermann. "Bitcoin and Beyond : A Technical Survey on Decentralized Digital Currencies." In: *IEEE COMMUNICATIONS SURVEYS & TUTORIALS* PP.99 (2015), pp. 1–1. ISSN: 1553-877X. DOI: doi:10.1109/COMST.2016.2535718.

[57] Xiaowei Xu *et al.* "Scan: a structural clustering algorithm for networks." In: *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2007, pp. 824–833.

[58] Kaizhong Zhang and Dennis Shasha. "Simple fast algorithms for the editing distance between trees and related problems." In: *SIAM journal on computing* 18.6 (1989), pp. 1245–1262.

부록 A Nebulas 계정 주소 설계

Nebulas 계정 주소는 "Nebulas"의 약자 인 "n"으로 시작하도록 설계했다.

A. 1 계정 주소

비트코인 및 이더리움과 마찬가지로 Nebulas는 Nebulas 계정의 기본 암호화 알고리즘으로 타원 곡선 알고리즘(elliptic curve algorithm)을 채택한다. 주소는 퍼블릭 키(public key)로부터 파생되며, 퍼블릭 키는 사용자의 패스프레이즈(passphrase)로 암호화된 프라이빗 키(private key)에서 파생된다.

또한 여러 가지 잘못된 문자가 입력되어 사용자가 실수로 NAS를 잘못된 사용자 계정으로 보내는 것을 방지하기 위한 검사합(Checksum) 설계를 갖췄다. 구체적인 계산식은 다음과 같다:

```
content = ripemd160(sha3_256(public key))
checksum = sha3_256(0x19<<(21*8) + 0x57<<(20*8) + content)[0:4]
address = base58(0x19<<(25*8) + 0x57<<(24*8) + content<<(4*8) + checksum)
```

0x19는 패딩(padding)용이며, 0x57은 주소 유형을 지정한다. `Content`의 길이는 20바이트이다. 검사합 길이는 4바이트이고 주소 길이는 26바이트이다.

Nebulas 주소는 base58로 인코딩되었으며 접두사(prefix) "n"을 포함하여 총 35자를 포함한다. 일반적인 주소는 다음과 같다: **`n1TV3sU6jyzR4rJ1D7jCAmtVGSntJagXZHC`**.

A. 2 스마트 컨트랙트 주소

스마트 컨트랙트 주소 계산은 계정 주소의 계산과 조금 다르다. 컨트랙트 송신자의 패스프레이즈(passphrase)가 필요하지 않으나, 송신자의 주소와 논스(nonce)가 필요하다. 계산식은 다음과 같다:

```
content = ripemd160(sha3_256(tx.from, tx.nonce))
checksum = sha3_256(0x19<<(21*8) + 0x58<<(20*8) + content)[0:4]
address = base58(0x19<<(25*8) + 0x58<<(24*8) + content<<(4*8) + checksum)
```

여기서 0x58은 스마트 계약의 주소 유형을 보여준다. 일반적인 스마트 계약 주소는 다음과 같다: **`n1sLnoc7j57YfzAVP8tJ3yK5a2i56QrTDdK`**.

부록 B 유사 스마트 컨트랙트 검색

코드 유사성의 어려움은 고급언어의 구조적 특징과 스마트 컨트랙트의 논리적 표현형태의 다양성에 있다. 현재, 일반적으로 학계에는 코드 유사성 알고리즘에 대한 다음과 같은 다양한 학파가 존재한다:

- **문자열 사이의 거리 편집(Edit Distance between Character Strings)**
  입력된 쿼리 코드와 후보 소스 코드는 모두 텍스트로 간주된다. 두 문자열 사이의 편집 거리는 두 문자열 사이의 유사성을 측정하는 데 사용된다. 편집 거리는 한 문자열을 다른 문자열로 변환하는 데 필요한 최소 편집 작업 수를 나타낸다. 허용된 편집 작업은 하나의 문자를 다른 문자로 대체, 즉 문자 삽입 및 문자 삭제를 포함한다. 일반적으로 편집 거리가 짧을수록 두 문자열 간의 유사도가 높아진다. 문자열 간 편집 거리를 기반으로 한 이 알고리즘은 소스 코드 비교뿐만 아니라 중간 표현 또는 기계어에까지 사용할 수 있다. 문자열 사이의 편집 거리에 기반한 알고리즘의 견고성을 향상시키기 위해 의미 변경 없이 소스 코드의 변환이 어느 정도 이루어진다. 예를 들어 공백 문자 제거, 주석 제거, '$'를 포함한 모든 로컬 변수 이름 바꾸기, 대수 표현의 정규 표현식 등이 이루어진다. 이 알고리즘은 속도가 빠르고, 간결하며, 효율이 높다는 점이 특징이다. 그러나 복잡한 프로그램에 대한 적응성은 상대적으로 열악하며, 코드의 구문 및 조직 구조를 고려하지 않는다.

- **토큰 시퀀스(Token Sequence)**
  토큰 시퀀스 표현 방법은 어휘 분석기의 분석을 통해 입력된 소스 코드를 토큰 시퀀스로 변환하는 것이다. 두 프로그램 간 유사성은 두 개의 토큰 시퀀스 사이의 유사성이므로 가장 긴 공통 부분 문자열 또는 상관 관계 일치 알고리즘(접미어 트리 매칭 알고리즘, suffix tree matching algorithm)을 사용하여 두 프로그램의 유사성을 측정할 수 있다. 이러한 과정에서 구문은 다르지만 기능이 유사한 코드 세그먼트가 감지될 수 있다. 그러나 이 방법은 두 프로그램 간의 유사성을 측정할 때 프로그램의 조직 구조를 감춘다.

- **추상 구문 트리(AST)**
  소스 코드의 구문 분석이 수행된 후, AST는 중간 표현식이 된다. 이 과정을 통해 하나의 하위 트리와 다른 하위 트리를 비교하여 두 프로그램 간의 유사성을 측정 할 수 있다. 두 트리 사이의 유사성을 측정하기 위해 트리 편집 거리 알고리즘[58]이 사용될 수 있다. 정확한 트리 편집 거리 알고리즘은 비교적 복잡하며 리터레처(Literature)[25]는 신속한 근사 알고리즘을 제공한다. 리터레처(Literature)[15]에 따르면, 구문 트리가 해시 지문(Hash fingerprint)에 종속되어 구문 트리 비교 알고리즘이 방대한 데이터 집합에 대해 고효율 검색을 수행할 수 있다.

- **프로그램 종속성 그래프(PDG)**
  PDG[18]는 프로그램의 내부 데이터를 표현하고 의존 관계를 통제하고 의미 수준에서 프로그램 코드를 분석 할 수 있다. 유사한 코드 프로토콜은 NP-완정성 문제이고 매우 복잡한 알고리즘을 필요로 하는 동형 서브 그래프(isomorphic subgraphs)가 검색 됨에 따라 근사 알고리즘만 사용 가능하다.

위에서 언급한 알고리즘은 다양한 차원에서 텍스트, 구조 및 구문 코드간의 유사성을 설명한다. 소스 약탈자(Source Forager)[27]는 훌륭한 엔지니어링 아이디어를 제공한다. 다양한 차원에서 유사성의 색인은 서로 다른 특징으로 묘사되며, 각각의 특징은 특정 측면에서의 코드 유사성 측정을 나타낸다. 마지막으로 벡터 유사성을 사용하여 전반적인 유사성 측정을 수행한다. 이 방법은 위에서 언급한 알고리즘의 장점을 통합한다. 이 아이디어는 Nebulas에서도 스마트 컨트랙트들 사이의 유사성 검색을 실현하는데 참고로 사용된다. Nebulas는 함수를 스마트 컨트랙트 사이에서 코드 검색의 기본 단위로 간주한다.

아래의 **표.3**은 후보 코드 유사성 특징을 정의하며 각 특징과 그 유사성을 계산하는 함수의 정의가 설명되어 있다:

표. 3: 코드 유사성 기능 패밀리 테이블

특징 분류	설명
유형 연산 결합(Type-Operation Coupling)	사용되는 유형과 수행되는 연산
스켈레톤 트리(Skeleton Tree)	루프와 조건의 구조
데코레이티드 스켈레톤(Decorated Skeleton)	루조 조건와 연산의 구조
3 Graph CGF BFS	크기 3의 CGF 하위 그래프, 하위 그래프 생성 시 사용되는 BGF
4 Graph CGF BFS	크기 4의 CGF 하위 그래프, 하위 그래프 생성 시 사용되는 BGF
3 Graph CGF DFS	크기 3의 CGF 하위 그래프, 하위 그래프 생성 시 사용되는 DFS
4 Graph CGF DFS	크기 4의 CGF 하위 그래프, 하위 그래프 생성 시 사용되는 DFS
라이브러리 호출(Library Calls)	라이브러리에 타입에 대한 호출
서명(Signature)	입력 유형 및 반환 유형
로컬 타입(Local Type)	로컬 변수 타입
숫자 리터럴(Numeric Literals)	사용된 숫자 데이터 상수
문자열 리터럴(String Literals)	사용된 문자열 데이터 상수

- **유형-연산 결합**
  이 특징은 2-튜플 집합이다. 두 개의 튜플은 변수 유형과 변수 유형의 연산자, 즉 (유형, 연산)쌍을 포함한다. 일반적으로 원시 데이터 유형은 산술 연산자, 논리 연산자, 관계 연산자 (예: (int, ≥))와 쌍을 이루어야 한다. 사용자 정의 데이터 유형(예: struct)은 (Bar, .foo)와 같은 멤버 함수와 쌍을 이루어야 한다. 그래서 데이터 유형 "Bar"의 필드 "foo"에 엑세스 되었다는 것을 나타낸다. 이 방법 기반으로 함수의 코드 본문에 있는 변수에 대한 모든 연산을 2 튜플로 변경할 수 있다.

  반복되는 내용을 제거한 후, 해당 코드 세그먼트의 유형-연산 결합(Type-Operation Coupling) 특성을 반영하기 위해 2 튜플 시퀀스가 사용된다. 우리는 비슷한 기능을 가진 코드가 유사한 변수 조작 세트를 가져야 한다고 생각한다. 그러나 우리는 2 튜플의 순서에 관심이 없으므로 이 기능은 코드의 논리 구조 정보를 잃어 버리게 되고, 따라서 코드의 기능만 부분적으로 나타낼 수 있다.

  유형-연산 결합 특징간의 유사성은 자코비안(Jacobian) 유사성으로 정의 할 수 있다. 즉, S1과 S2의 두 세트가 주어지면 자코비안 유사성은 다음 공식으로 정의 될 수 있다:

$$sim_{Jacc(S_1,S_2)} = \frac{|\ S_1 \bigcap S_2\ |}{|\ S_1 \bigcup S_2\ |} \tag{12}$$

- **스켈레톤 트리(Skeleton tree)**
  코드 기반 추상 구문 트리이다. 그러나 루프(for, while, do .. while)와 조건문(if .. else)만이 보유 되고 다른 모든 노드는 트리에서 제거된다. Nebulas는 비슷한 함수를 가진 코드는 루프와 조건문의 구조가 비슷해야 한다고 생각한다. 스켈레톤 트리의 유사도 계산은 두 트리 사이의 편집 거리를 기반으로 한다. d_r은 두 트리 사이의 예상 편집 거리로 정의되며 트리의 크기만이 결정의 변수로 작용한다. 즉 :

$$d_r(T_1, T2) = \frac{|\ size(T_1) - size(T_2)\ |}{max(size(T_1), size(T_2))} \tag{13}$$

D_T는 편집 거리의 임계값으로 가정되어 0.5로 설정된다. 두 트리 사이에서 대략적인 편집 거리 계산 공식을 추가적으로 획득 할 수 있다:

$$d_t(T_1, T2) = \begin{cases} d_r(T_1, T_2) & if\ d_r(T_1, T_2) \geq D_r \\ \dfrac{max\begin{pmatrix} ed(pre(T_1),\ pre(T_2)), \\ ed(post(T_1),\ post(T_2)) \end{pmatrix}}{max(size(T_1), size(T_2))} & otherwise \end{cases} \tag{14}$$

pre(T)는 트리의 전위 운행 시퀀스를 나타낸다. post(T)는 트리의 후위 운행 시퀀스를 나타낸다. $ed(S_1, S_2)$는 S_1과 S_2 사이의 편집 거리를 나타낸다. 두 개의 스켈레톤 트리 사이의 유사성은 다음 공식을 사용하여 계산할 수 있다 :

$$sim_{Tree}(T_1, T2) = 1 - d_t(T_1, T2) \tag{15}$$

- **데코레이티드 스켈레톤 트리(Decorated Skeleton Tree)**
  데코레이티드 스켈레톤 트리는 스켈레톤 트리와 유사하다. 루프 및 분기 노드 외에도 대부분의 연산자(예 : +, -, <)를 보유한다. 그러나 대부분의 연산자가 노이즈이기 때문에 지정 연산자가 제거된다.

- **CFG의 K-서브그래프**
  함수의 CFG의 k-서브 그래프를 기반으로 실현된 k-서브 그래프는 다음과 같이 정의되어야 한다:

  CFG와 특정 노드가 주어져야 한다. 이를 기반으로 형성된 서브 그래프가 k-서브 그래프가 되어야 할 때, 횡단 노드의 수가 k에 도달 할 때까지 너비 우선 검색 (BFS) 또는 깊이 우선 검색(DFS)을 수행해야 한다. 순회가 끝난 후 노드 수가 k에 도달하지 못하면 해당 서브 그래프를 삭제해야 한다. CFG의 각 노드 횡보를 통해 모든 k-서브 그래프를 획득 할 수 있다.
  각 k-서브그레프에 대해 k^2 비트 정수가 사용되어 표현한다. (**그림.20**을 참조) 모든k-서브 그래프는 하나의 정수 집합을 형성한다.

  3 Graph CFG BFS: k = 3, BFS Traversal
  4 Graph CFG BFS: k = 4, BFS Traversal
  3 Graph CFG DFS: k = 3, DFS Traversal
  4 Graph CFG DFS: k = 4, DFS Traversal

  유사성은 일반 자코비안 유사도 공식으로 계산할 수 있다. $\overrightarrow{x} = (x_1, x_2, \ldots x_n)$ 와 $\overrightarrow{y} = (y_1, y_2, \ldots y_n)$ 벡터는 다음과 같이 정의 될 수 있는 일반 자코비안 유사성을 기반으로 주어진다:

$$J(\overrightarrow{x}, \overrightarrow{y}) = \frac{\sum_i min(x_i, y_i)}{\sum_i max(x_i, y_i)} \tag{16}$$

	A	B	C	D
A	0	1	0	0
B	0	0	1	1
C	0	0	0	0
D	0	0	0	0

그림. 20: 4-그래프 예시: 인접 행렬의 요소는 10 진수가 17152인 이진 문자열 "0100 0011 0000 0000”이다

- **라이브러리 호출**
  컨트랙트에 다른 라이브러리 컨트랙트가 있는 경우, 모든 라이브러리 컨트랙트의 주소가 기록된다. 주소 사이의 유사성은 자코비안(Jacobian) 유사성 공식으로 계산된다.

- **타입 서명**
  이 특징은 입력 매개 변수 유형과 반환 매개 변수 유형으로 구성되며 이들 사이의 유사성은 자코비안(Jacobian) 유사성 공식으로 계산할 수 있다. 예를 들어 다음 스마트 컨트랙트 코드의 경우 "getBalance"함수의 타입 서명 특징은 vector (address, uint256)이다.

```
contract addressTest {
   function getBalance(address addr) returns (uint) {
          return addr.balance;
  }
}
```

- **로컬 타입**
  해당 특징은 함수 몸체의 지역 변수의 모든 유형 집합이며, 자코비안(Jacobian) 유사도 공식으로 유사도를 계산해야 한다.

- **숫자 리터럴**
  모든 숫자 상수 집합은 숫자 리터럴의 특징으로 활용되며 자코비안(Jacobian) 유사도 공식으로 유사도를 계산해야 한다.

- **문자열 리터럴**
  모든 문자열 상수 집합은 문자열 리터럴의 특징으로 활용되며 자코비안(Jacobian) 유사도 공식으로 유사도를 계산해야 한다.

피처 패밀리(Feature family)를 확장 할 수 있어, 편리하게 새로운 특성을 추가할 수 있다. 각 특성에 대한 유사도 계산이 존재하는 상황을 바탕으로 모든 특성의 가중치 합을 계산하므로 최종 코드 유사도를 구할 수 있다.

$$sim_{combined}(\overrightarrow{A}, \overrightarrow{B}) = \frac{\sum_{c=1}^{n_{cl}} sim_c(\overrightarrow{A_c}, \overrightarrow{B_c}) \cdot w_c}{\sum_{c=1}^{n_{cl}} w_c} \tag{17}$$

여기서 $\overrightarrow{A}$와 $\overrightarrow{B}$는 고유 벡터이다. n_{cl}은 피처 패밀리(Feature family)의 특성 수이다. sim_c는 특성 c에 특정한 유사도 계산 함수이다. $\overrightarrow{A_c}$와 $\overrightarrow{B_c}$는 특성 c의 고유 벡터이다. w_c는 c의 가중치이다. 가중치는 다수의 테스트 집합을 기반으로 한 머신 러닝 알고리즘을 통해 획득 할 수 있다.