(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2021-0080818
(43) 공개일자 2021년07월01일

(51) 국제특허분류(Int. Cl.)
*H04L 9/08* (2006.01) *H04L 29/08* (2006.01)
*H04L 9/06* (2006.01)
(52) CPC특허분류
*H04L 9/0838* (2013.01)
*H04L 67/1051* (2013.01)
(21) 출원번호 10-2019-0172804
(22) 출원일자 2019년12월23일
심사청구일자 2019년12월23일

(71) 출원인
**서강대학교산학협력단**
서울특별시 마포구 백범로 35 (신수동, 서강대학교)
**주식회사 엠블럭**
서울특별시 마포구 와우산로5길 17, 1층(상수동)
(72) 발명자
**서중원**
서울특별시 서대문구 증가로26길 18-5 (북가좌동) 지층 우측
**고덕윤**
서울특별시 영등포구 선유서로 67 신동아하이팰리스 102동 601호
**박수용**
서울특별시 마포구 독막로 18 길 5, 101동 1001호 (상수동, 상수두산위브아파트)
(74) 대리인
**이지연**

전체 청구항 수 : 총 12 항

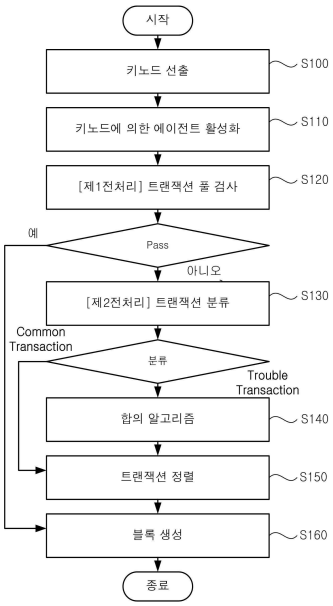
(54) 발명의 명칭 **에이전트 기반의 블록체인 네트워크 및 상기 네트워크의 전처리 방법**

## (57) 요 약

본 발명은 에이전트 기반의 블록체인 네트워크에서의 전처리 방법에 관한 것이다. 상기 전처리 방법은, (a) 상기 복수 개의 노드들 중 하나의 키노드를 선출하는 단계; (b) 상기 키노드에 의해 에이전트 노드를 검증하고 활성화시키는 단계; (c) 상기 복수 개의 노드들이 상기 에이전트 노드에게 트랜잭션 풀(Transaction Pool)을 제공하는

(뒷면에 계속)

*대 표 도* - 도1



단계; (d) 상기 에이전트 노드는 모든 노드들의 트랜잭션 풀을 검사하는 단계; (e) 상기 에이전트 노드는 모든 노드들의 트랜잭션 풀이 같으면, 키노드에게 검증받고 키노드의 서명과 함께 결과값을 상기 복수 개의 노드들에게 전송하여 합의 알고리즘없이 블록을 생성하는 단계; (f) 상기 에이전트 노드는 트랜잭션 풀이 같지 않으면, 공통 트랜잭션과 근심 트랜잭션을 분류하고 근심 트랜잭션에 대하여 합의 알고리즘을 적용하여 합의된 트랜잭션을 분류하는 단계; (g) 상기 공통 트랜잭션과 상기 합의된 근심 트랜잭션에 대하여 블록을 생성하는 단계;를 구비한다.

(52) CPC특허분류
***H04L 67/1093*** (2013.01)
***H04L 9/0643*** (2013.01)
***H04L 9/0819*** (2013.01)
***H04L 9/0869*** (2013.01)
*H04L 2209/38* (2013.01)

이 발명을 지원한 국가연구개발사업

| | |
|---|---|
| 과제고유번호 | 1711093140 |
| 부처명 | 과학기술정보통신부 |
| 과제관리(전문)기관명 | 정보통신기획평가원 |
| 연구사업명 | 대학ICT연구센터육성지원사업 |
| 연구과제명 | 적응형 블록체인 플랫폼 기술 개발 및 전문 인력 양성 |
| 기 여 율 | 1/1 |
| 과제수행기관명 | 서강대학교산학협력단 |
| 연구기간 | 2019.03.01 ~ 2019.12.31 |

# *명 세 서*

## *청구범위*

### 청구항 1

복수 개의 노드들을 구비하는 블록체인 네트워크에 있어서,

블록 생성하기 전에 전처리를 수행하는 에이전트 노드; 를 더 구비하고,

상기 블록체인 네트워크에 구비된 복수 개의 노드들 중 하나는 키 노드로 선출되며,

상기 에이전트 노드는 상기 키노드에 의해 검증된 후 상기 키노드로부터 서명과 공개키를 전송받아 활성화되며,

상기 활성화된 에이전트 노드는 복수 개의 노드들이 각각 갖고 있는 트랜잭션 풀(transaction pool)이 모두 동일한지 확인하고 그 결과에 따라 합의 알고리즘의 수행없이 블록 생성할지 여부를 결정하는 것을 특징으로 하는 블록체인 네트워크.

### 청구항 2

제1항에 있어서, 상기 에이전트 노드는,

상기 블록체인 네트워크에 포함된 복수 개의 노드들과는 분리되어 동작되는 것을 특징으로 하는 블록체인 네트워크.

### 청구항 3

제1항에 있어서, 상기 복수 개의 노드들은 상기 활성화된 에이전트 노드에게 트랜잭션 풀을 모두 제공하고,

상기 에이전트 노드는 모든 노드의 트랜잭션 풀이 동일하면, 키노드에게 검증받고, 키노드의 서명과 함께 결과값을 상기 복수 개의 노드들에게 전송하고,

상기 복수 개의 노드들은 상기 에이전트 노드으로부터 전송받은 결과값을 기반으로 하여 합의 과정없이 블록을 생성하는 것을 특징으로 하는 블록체인 네트워크.

### 청구항 4

제1항에 있어서, 상기 복수 개의 노드들은 상기 활성화된 에이전트 노드에게 트랜잭션 풀을 모두 제공하고,

상기 에이전트 노드는 모든 노드의 트랜잭션 풀이 동일하지 않으면, 트랜잭션들을 분류하여, 모든 노드가 동일하게 갖고 있는 트랜잭션은 공통 트랜잭션(Common Transaction)으로 분류하고, 공통 트랜잭션은 블록이 생성될 때까지 에이전트 노드가 임시 저장하고, 공통 트랜잭션 외의 트랜잭션은 근심 트랜잭션(Trouble transaction)으로 분류하는 것을 특징으로 하는 블록체인 네트워크.

### 청구항 5

제4항에 있어서, 상기 에이전트 노드는,

상기 근심 트랜잭션에 대하여 사전 설정된 합의 알고리즘을 적용하여 합의하고, 합의가 끝난 근심 트랜잭션은 합의된 트랜잭션으로 분류하는 것을 특징으로 하는 블록체인 네트워크.

### 청구항 6

제5항에 있어서, 상기 에이전트 노드는,

임시 저장된 공통 트랜잭션과 근심 트랜잭션 중 상기 합의된 트랜잭션을 하나의 임시 저장소에 저장하여 정렬을 진행하고, 그 결과값을 키노드에게 전송하여 검증받고, 키노드의 서명과 함께 결과값을 상기 복수 개의 노드들에게 전송하는 것을 특징으로 하며,

상기 복수 개의 노드들은 상기 에이전트 노드으로부터 키노드의 서명과 함께 전송받은 결과값을 기반으로 하여

합의 과정없이 블록을 생성하는 것을 특징으로 하는 블록체인 네트워크.

**청구항 7**

제1항에 있어서, 상기 블록체인 네트워크에 포함된 복수 개의 노드들의 각각은,

순차적으로, 범위 설정후, 블록의 해시값을 시드로 사용하여 랜덤 시드 알고리즘의 결과값을 설정하고, 상기 결과값을 이용하여 상기 복수 개의 노드들 중 하나의 키 노드를 선출하는 것을 특징으로 하는 블록체인 네트워크.

**청구항 8**

복수 개의 노드들과 에이전트 노드를 구비하는 블록체인 네트워크에 있어서,

(a) 상기 복수 개의 노드들 중 하나의 키노드를 선출하는 단계;

(b) 상기 키노드에 의해 에이전트 노드를 검증하고, 키노드의 검증이 성공적으로 완료되면 에이전트 노드에게 서명과 공개키를 전송하여 에이전트 노드를 활성화시키는 단계;

(c) 상기 키노드의 상기 에이전트 노드에 대한 검증이 성공적으로 완료되면 에이전트가 활성화되었음을 상기 복수 개의 노드들에게 알리는 단계;

(d) 상기 복수 개의 노드들이 상기 에이전트 노드에게 트랜잭션 풀(Transaction Pool)을 제공하는 단계;

(e) 상기 에이전트 노드는 모든 노드들의 트랜잭션 풀을 검사하는 단계;

(f) 상기 에이전트 노드는 모든 노드들의 트랜잭션 풀이 같으면, 키노드에게 검증받고 키노드의 서명과 함께 결과값을 상기 복수 개의 노드들에게 전송하는 단계;

(g) 상기 복수 개의 노드들이 에이전트 노드로부터 키노드의 서명을 포함한 결과값을 기준으로 하여 합의 알고리즘없이 블록을 생성하는 단계;

를 구비하는 것을 특징으로 하는 에이전트 기반의 블록체인 네트워크에서의 전처리 방법.

**청구항 9**

제8항에 있어서, 상기 블록체인 네트워크에서의 전처리 방법은,

(h) 상기 (e) 단계에서 모든 노드들의 트랜잭션 풀이 같지 않으면, 트랜잭션들을 분류하여 합의가 필요한 트랜잭션을 찾는 단계;를 더 구비하고,

상기 (h) 단계는, 모든 노드가 동일하게 갖고 있는 트랜잭션은 공통 트랜잭션(Common Transaction)으로 분류하고, 공통 트랜잭션은 블록이 생성될 때까지 에이전트 노드가 임시 저장하고, 공통 트랜잭션 외의 트랜잭션은 근심 트랜잭션(Trouble transaction)으로 분류하는 것을 특징으로 하는 에이전트 기반의 블록체인 네트워크에서의 전처리 방법.

**청구항 10**

제9항에 있어서, 상기 블록체인 네트워크에서의 전처리 방법은,

(i) 상기 근심 트랜잭션에 대하여 사전 설정된 합의 알고리즘을 적용하여 합의하고, 합의가 끝난 근심 트랜잭션은 합의된 트랜잭션으로 분류하는 단계; 를 더 구비하는 것을 특징으로 하는 에이전트 기반의 블록체인 네트워크에서의 전처리 방법.

**청구항 11**

제10항에 있어서, 상기 블록체인 네트워크에서의 전처리 방법은,

(j) 상기 에이전트 노드는 상기 (h) 단계에서 임시 저장된 공통 트랜잭션과 상기 (i) 단계에서 상기 합의된 트랜잭션을 하나의 임시 저장소에 저장하여 정렬을 진행하고, 그 결과값을 키노드에게 전송하여 검증받고, 키노드의 서명과 함께 결과값을 상기 복수 개의 노드들에게 전송하는 단계;를 더 구비하는 것을 특징으로 하는 에이전트 기반의 블록체인 네트워크에서의 전처리 방법.

**청구항 12**

제8항에 있어서, 상기 (a) 단계는,

블록체인 네트워크에 포함된 복수 개의 노드들에게, 범위 설정후, 블록의 해시값을 시드로 사용하여 랜덤 시드 알고리즘의 결과값을 설정하고, 상기 결과값을 이용하여 상기 복수 개의 노드들 중 하나의 키 노드를 선출하는 것을 특징으로 하는 에이전트 기반의 블록체인 네트워크에서의 전처리 방법.

## 발명의 설명

### 기 술 분 야

[0001] 본 발명은 블록체인 네트워크에 관한 것으로서, 더욱 구체적으로는 블록체인 네트워크의 확장성을 위하여 에이전트를 사용하여 전처리함으로써 합의 알고리즘의 적용없이 블록을 생성할 수 있도록 하여 합의에 따른 처리 속도를 향상시킬 수 있도록 한 에이전트 기반의 블록체인 네트워크 및 상기 네트워크에서의 전처리 방법에 관한 것이다.

### 배 경 기 술

[0002] 블록체인이란 데이터 분산 저장 기술로서, 모든 사용자가 네트워크에서 발생하는 모든 데이터를 각자 저장하여, 데이터의 무결성과 투명성을 유지할 수 있게 만든 기술이다. 검증된 데이터들은 하나의 블록 안에 저장이 되고, 그 블록들은 앞뒤로 연결이 되어 네트워크에 저장되는 형태이기 때문에 블록체인이라 불리게 되었다. 따라서, 블록체인 네트워크는 중앙 관리 기관 없이 사용자들이 직접 거래할 수 있게 된다. 블록체인은 트랜잭션의 위조 및 변조가 어려우므로 데이터의 신뢰성을 보장받을 수 있을 뿐만 아니라 모든 사용자들이 동일한 블록체인 원장을 보유함으로써 누구나 열람할 수 있기 때문에 데이터의 투명성과 무결성을 보장하게 된다.

[0003] 이러한 블록체인 네트워크는 퍼블릭(Public) 블록체인 네트워크와 프라이빗(Private) 블록체인 네트워크로 나뉘게 된다. 퍼블릭 블록체인 네트워크는 모든 데이터가 공개되며 누구나 네트워크에 참여하여 데이터를 검증하고 블록을 생성할 수 있는 노드가 될 수 있으며, 현재 비트코인 및 이더리움이 퍼블릭 블록체인 네트워크로 이루어진다.

[0004] 한편, 프라이빗 블록체인 네트워크는 기업을 위하여 개발되었으며, 사적 블록체인, 폐쇄형 블록체인, 허가형 블록체인, 기업형 블록체인 또는 엔터프라이즈 블록체인이라고도 불리우며, 허가된 참여자 외의 거래 내역과 여러 행동(Action)은 공유되지 않고 추적할 수 없다. 그리고, 프라이빗 블록체인 네트워크에 참여하기 위하여, 한 명의 주체로부터 허가된 참여자만이 참여하여 블록을 생성할 수 있으며, 프라이빗 블록체인의 대표적인 사례로는 하이퍼레저 패브릭(Hyperledger Fabric)이 있다. 하이퍼레저 패브릭은 허락된 참여자만 블록체인 네트워크에 참여 가능하며 네트워크내에서 조직(Organization)을 구성하고 논리적 묶음(Channel)으로 서로 다른 원장의 공유가 가능하다.

[0005] 블록체인을 개발하는 데 있어서 다양한 기술적 요소들을 블록체인의 특징에 맞춰서 적용할 수 있지만, 블록체인에 필수적으로 적용되는 핵심 기술은 아래와 같이 세 가지로 정리할 수 있다.

[0006] ① Peer to Peer (P2P)

[0007] ② 네트워크암호학

[0008] ③ 합의 알고리즘

[0009] P2P 네트워크는 블록체인 네트워크가 데이터를 분산하여 저장할 수 있도록 하는 블록체인의 가장 근본이 되는 기술이다. 기존 클라이언트 서버 구조에서는 모든 데이터를 서버에서 처리하기 때문에 데이터가 서버에 집중되지만, 블록체인에서는 P2P 네트워크를 통해 한 명에게 데이터가 집중되는 현상을 제거하고, 참여자들 간에 동등하게 데이터를 갖고 통신할 수 있도록 보장한다.

[0010] 블록체인의 두 번째 기술인 암호학 기술은 블록체인에서 참여자들 간의 서로 데이터를 주고받는 통신 과정에서의 안전성 및 블록체인 내부 데이터의 검증을 위해 사용한다. 대칭 키 알고리즘을 사용하여 블록체인 내부 사용자들의 부인 방지와 데이터 전송과정에서 발생할 수 있는 해킹의 위협을 막는다. 또한, 블록체인 내부 데이터의 검증을 위해서 해시 알고리즘과 개인키를 이용한 서명알고리즘을 사용한다. 마지막으로 핵심 기술은 합의 알고리즘이다. 합의 알고리즘은 서버가 없는 블록체인 구조에서 참여자들 간의 같은 데이터를 가질 수 있도록 보장

하는 기술이다.

[0011] 한편, 블록체인의 합의 알고리즘에 대하여 살펴본다. 합의 알고리즘이 없으면 서버가 없는 P2P 네트워크의 블록체인 노드들이 같은 데이터를 갖도록 보장할 수 없기 때문에 합의 알고리즘은 반드시 필요하지만, 합의 알고리즘에 의해서 블록체인의 확장성과 처리 속도가 영향을 받는다. 오늘날에는 다양한 분야에서 블록체인의 확장성에 영향을 주지 않으면서 처리 속도를 빠르게 만들기 위한 합의 알고리즘 연구와 개발이 지속해서 이루어지고 있다. 공개된 합의 알고리즘 중 가장 잘 알려진 합의 알고리즘은 작업증명, 지분증명, 위임 지분 증명 방식이 있다.

[0012] 작업증명 방식은 경쟁을 통해 블록생성 권한을 주는 방식이다. 작업증명은 단방향 해시 알고리즘의 연산을 가장 빠르게 연산한 참여자 한 명에게 블록 생성 권한을 주는 방식을 사용한다. 하지만 해시 알고리즘의 연산을 빠르게 하기 위해서는 일정 수준 이상의 장비가 필요하여 전력 낭비가 많이 소모된다는 단점이 있다.

[0013] 지분증명 방식은 경쟁이 심한 작업증명 방식의 문제를 해결하기 위해 개발된 방식으로 네트워크에서 정의한 지분에 따라서 노드들은 각기 다른 지분을 가지게 되고 노드가 가진 지분에 따라서 작업증명의 난이도를 변경시켜주는 방식이다. 하지만, 지분증명의 경우 특정 노드가 계속해서 블록을 생성하게 되는 블록 생성의 중앙화 문제가 있다.

[0014] 위임 지분 증명 방식은 미국의 선거인단 제도와 같이 블록체인 네트워크에서 노드들이 대표자를 선출한다. 대표자 수는 미리 고정되어 있으며, 선출된 대표자들끼리 투표를 통하여 블록을 생성한다. 위임 지분 증명 방식은 대표자가 누가 될지 예측할 수 있어 대표자가 외부 공격을 받을 위협이 있다는 단점이 있다.

[0015] 퍼블릭 블록체인의 경우 익명의 사용자들이 모여서 네트워크를 유지하기 때문에 블록체인 네트워크 안에서 한 사람 혹은 그룹을 지목해서 블록 생성 권한을 줄 경우 블록체인 네트워크의 신뢰도가 낮아질 수 있다. 그래서 퍼블릭 블록체인에서는 노드들의 경쟁을 통해, 블록을 생성할 한 명의 사용자를 선출하여 보상을 주는 합의 알고리즘을 선호한다.

[0016] 프라이빗 블록체인은, 퍼블릭 블록체인과 달리, 공동의 목적을 가진 신원이 명확한 사용자들만 블록체인 네트워크에 참여하므로 서로 간의 경쟁을 통하여 블록을 생성할 한 명의 사용자를 선출하는 방식을 선호하지 않는다. 프라이빗 블록체인은 데이터의 안정성(Safety)을 우선시하는 투표 방식의 합의 알고리즘 방식을 주로 선호한다. 특히, 비잔틴 장애 허용(BFT: Byzantine Fault Tolerance) 방식의 합의 알고리즘을 주로 사용한다. 하지만 BFT 방식의 합의 알고리즘은 모든 참여자가 합의에 참여하는 구조로 되어있어서 참여 노드가 많아지면 처리속도가 감소하는 확장성의 문제가 있다. 실용적 비잔틴 장애 허용 (PBFT: Practical Byzantine Fault Tolerance)은 BFT 알고리즘을 비동기 방식으로 접근하여 확장성을 개선한 대표적인 알고리즘이지만, PBFT 알고리즘 또한 참여자 수에 증가함에 따라 처리 속도가 현격히 감소하는 문제가 있다

[0017] 분산 네트워크와 블록체인 분야에서 BFT 알고리즘의 확장성을 개선하기 위한 연구는 지속해서 진행되어 왔다. 분산 네트워크 환경에서의 BFT 확장성 개선을 위한 연구는 BFT-SMART, Zyzzyva, SBFT이 있으며, SCP, Multi-Agent, HotStuff의 연구들은 블록체인 환경에서의 BFT 확장성 개선을 위한 연구를 진행하였다.

[0018] BFT-SMART는 합의 과정이 총 3단계인 Propose, Write, Accept 과정으로 구성되어 있고, 각각의 단계마다 모든 노드가 투표에 참여하여 합의를 진행한다. BFT-SMART의 가장 큰 특징은 노드 간의 충돌이나 시스템 강제적 종료 상황 등을 고려하지 않고 합의를 진행하던 기존 합의 방식에서 벗어나 전처리 과정을 통해서 노드의 강제적 종료 상황 등을 먼저 파악한다. BFT-SMART는 정상적으로 작동하는 노드의 수가 1/2 이상일 경우 합의 허용률을 2/3로 하여 합의를 진행한다. 만약 정상적으로 작동하는 노드의 수가 1/2 이하일 경우 작동하지 않는 노드들은 시스템을 재 시작한다. 시스템을 재시작한 노드들은 정상적으로 작동하고 있는 노드들에 데이터를 전송받아 이전 데이터를 복구하고 다시 합의 과정에 참여하게 된다. BFT-SMART는 노드의 강제적 종료 상황 등의 충돌을 사전에 확인하여 합의 과정에서 발생할 수 있는 장애를 미리 방지하여 합의의 확장성을 개선하였다. 하지만 BFT-SMART의 전처리 과정은 시스템 충돌을 미리 방지하는 과정에 불과하므로, 시스템 충돌이 발생하지 않는 이상 처리 속도를 개선하지 못한다는 한계가 있다.

[0019] Zyzzyva 연구는 노드들끼리의 합의가 실패하는 경우 합의를 처음부터 다시 진행하지 않고 동기화를 진행 후, 합의 허용 범위를 높여서 합의를 계속 진행하는 방식이다. Zyzzyva의 동작 구조는 2/3 이상의 합의 규칙하에서 합의에 도달하지 못하면, 데이터 동기화를 진행한 뒤 합의에 장애를 일으킨 노드를 제외하고, 합의 허용 범위를 1/2로 낮추어 합의를 진행한다. 하지만, 이는 일시적으로 장애 허용률을 높이게 되어 악의적 공격으로부터 노출될 수 있다는 문제점이 있으며, 동기화를 진행하는 노드가 장애가 발생하면 합의에 도달하지 못하게 될 수 있는

위험이 있다.

[0020] SBFT(State of the art Byzantine Fault Tolerant)는 기존 BFT 알고리즘 계열의 합의가 한번 시작한 경우 합의가 완료될 때까지 계속해서 통신을 진행하여 통신량에 의해 합의의 확장성이 낮아지는 문제를 해결하기 위한 연구를 진행하였다. SBFT는 합의 중간에 각기 다른 두 개의 노드를 사용하여 두 번의 동기화를 진행한다. 동기화를 통해 사전에 잘못된 데이터를 파악하고, 합의 과정에서 발생하는 통신량을 줄여 확장성을 개선하는 연구를 진행하였다. SBFT의 연구는 동기화를 진행하는 노드가 장애가 발생하면 동기화에 실패하게 되어 결국 합의에 도달하지 못하게 될 가능성이 있음으로 합의의 지속성(Liveness)이 보장되지 못한다.

[0021] SCP(Scalable Consensus Protocol)는 기존 합의 알고리즘인 작업증명과 PBFT 합의 알고리즘을 융합하는 방법을 사용하였으며, 전처리 과정을 통해서 노드의 그룹화를 통해 합의 과정에 참여하는 노드의 수를 제한하여 확장성을 개선하였다. 전처리 과정으로써 각각의 노드는 작업증명을 수행하게 된다. 작업증명을 통해 획득한 논스(nonce)값을 통해 노드들은 하나의 그룹들로 엮이게 된다. 각각의 그룹들은 PBFT를 활용하여 합의를 진행하게 되고, 그룹의 합의가 종료된 이후에 그룹별 결과를 취합한다. SCP의 경우 각 노드의 작업증명을 통해 그룹을 생성하는 시간과 각각의 그룹들의 합의 결과를 취합하는 시간이 길어지면 처리 속도가 더 늦어지게 되는 문제점이 있다.

[0022] Multi-Agent의 경우 SCP 연구와 비슷하게 전처리를 통해서 노드의 그룹화를 통해 합의 과정에 참여하는 노드의 수를 제한한다. 하지만, Multi-Agent의 경우 SCP와 달리 네트워크 토폴로지를 조정을 통해 노드 그룹을 생성한다. 각각의 그룹의 대표는 에이전트라 표현이 되고, 각각의 그룹의 합의가 종료되었을 때 에이전트끼리 합의 결과를 취합한다. Multi-Agent 연구도 SCP처럼 그룹 생성을 위해서 네트워크 토폴로지를 조정하는 시간이나, 그룹의 합의 결과를 취합하는 시간이 길어질 경우 처리 속도가 더 늦어지게 된다.

[0023] 기존 BFT의 확장성을 개선한 연구들은 통신 방법을 비동기 방식 (Asynchronous)을 사용하여 확장성을 개선하였다. 하지만, HotStuff의 경우 부분 동기화(Partial Synchrony) 방식을 사용한다. 부분 동기화란 동기 방식(Synchronous)과 비동기 방식을 융합한 것으로, 정해진 일정 시간 동안(GST, Global Standard Time)은 비동기 방식을 사용하여 통신을 진행한다. GST이후, 비동기 통신에서 제외된 노드들만 동기화 방식의 통신을 진행한다. 또한, HotStuff는 임계치 서명(Threshold Signature) 방식을 사용한다. 기존 합의 알고리즘에서는 서명을 검증하기 위해서는 노드들이 각자 받은 서명을 함께 받은 전송자의 공개키를 활용하여 검증을 진행한다. 하지만 임계치 서명 방식은 서명을 검증하기 위해서 미리 설정된 수만큼의 서명을 받으면 해당 서명을 더 검증하지 않는 방식이다. 예를 들어 전체 블록체인 네트워크 노드가 40개 일 때, 임계치 서명의 임계치를 10개의 노드로 지정해 놓았다면, 10개의 노드가 해당 서명 검증을 완료 후에는 나머지 30명의 노드는 해당 서명을 검증하지 않아도 되는 방식을 이야기한다. 부분 동기화 방식과 임계치 서명 방식을 활용하면 노드가 받는 데이터의 양을 최소화할 수 있다.

[0024] HotStuff에서는 노드가 받는 데이터양을 최소화하여 합의 과정에서 멀티 스레드(Multi-Thread)를 적용하여 동시에 각기 다른 합의를 동시에 진행할 수 있는 방식을 제안하여 확장성을 개선하였다. 하지만 노드가 증가함에 따라서 멀티 스레드 방식으로 노드들이 받는 부하가 점점 심해진다는 한계가 있다.

[0025] 향후 전 세계적으로 블록체인 시장의 크기가 점점 증가할 것으로 고려하면, 참여자가 많은 프라이빗 블록체인 환경에서도 BFT 알고리즘을 적용하기 위해서는 확장성 개선이 필수적이다. 따라서, 본 발명에서는 프라이빗 블록체인에서 노드끼리 합의 없이 데이터를 기록하다가 에이전트가 합의가 필요한 순간을 알려주는 전처리 기법을 제안하고자 한다.

## 선행기술문헌

### 특허문헌

[0026] (특허문헌 0001) 한국공개특허공보 제 10-2019-0088530호

(특허문헌 0002) 한국등록특허공보 제 10-1975822호

(특허문헌 0003) 한국등록특허공보 제 10-2050087호

## 발명의 내용

### *해결하려는 과제*

[0027] 전술한 문제점을 해결하기 위한 본 발명의 목적은 많은 참여 노드가 있는 프라이빗 블록체인 네트워크에서 비잔틴 장애 허용을 적용할 수 있도록 하기 위하여, 합의없이 데이터를 기록하다가 에이전트가 합의가 필요한 순간을 알려주도록 하는 에이전트 기반의 블록체인 네트워크에서의 전처리 방법 및 블록체인 네트워크를 제공하는 것이다.

### *과제의 해결 수단*

[0028] 전술한 목적을 달성하기 위한 본 발명의 제1 특징에 따른 복수 개의 노드들을 구비하는 블록체인 네트워크는, 블록 생성하기 전에 전처리를 수행하는 에이전트 노드; 를 더 구비하고,

[0029] 상기 블록체인 네트워크에 구비된 복수 개의 노드들 중 하나는 키 노드로 선출되며, 상기 에이전트 노드는 상기 키노드에 의해 검증된 후 상기 키노드로부터 서명과 공개키를 전송받아 활성화되며, 상기 활성화된 에이전트 노드는 복수 개의 노드들이 각각 갖고 있는 트랜잭션 풀(transaction pool)이 모두 동일한지 확인하고 그 결과에 따라 합의 알고리즘의 수행없이 블록 생성할지 여부를 결정한다.

[0030] 전술한 제1 특징에 따른 블록체인 네트워크에 있어서, 상기 에이전트 노드는, 상기 블록체인 네트워크에 포함된 복수 개의 노드들과는 분리되어 동작되는 것이 바람직하다.

[0031] 전술한 제1 특징에 따른 블록체인 네트워크에 있어서, 상기 복수 개의 노드들은 상기 활성화된 에이전트 노드에게 트랜잭션 풀을 모두 제공하고, 상기 에이전트 노드는 모든 노드의 트랜잭션 풀이 동일하면, 키노드에게 검증받고, 키노드의 서명과 함께 결과값을 상기 복수 개의 노드들에게 전송하고, 상기 복수 개의 노드들은 상기 에이전트 노드로부터 전송받은 결과값을 기반으로 하여 합의 과정없이 블록을 생성하는 것이 바람직하다.

[0032] 전술한 제1 특징에 따른 블록체인 네트워크에 있어서, 상기 복수 개의 노드들은 상기 활성화된 에이전트 노드에게 트랜잭션 풀을 모두 제공하고, 상기 에이전트 노드는 모든 노드의 트랜잭션 풀이 동일하지 않으면, 트랜잭션들을 분류하여, 모든 노드가 동일하게 갖고 있는 트랜잭션은 공통 트랜잭션(Common Transaction)으로 분류하고, 공통 트랜잭션은 블록이 생성될 때까지 에이전트 노드가 임시 저장하고, 공통 트랜잭션 외의 트랜잭션은 근심 트랜잭션(Trouble transaction)으로 분류하는 것이 바람직하다.

[0033] 전술한 제1 특징에 따른 블록체인 네트워크에 있어서, 상기 에이전트 노드는, 상기 근심 트랜잭션에 대하여 사전 설정된 합의 알고리즘을 적용하여 합의하고, 합의가 끝난 근심 트랜잭션은 합의된 트랜잭션으로 분류하는 것이 바람직하다.

[0034] 전술한 제1 특징에 따른 블록체인 네트워크에 있어서, 상기 에이전트 노드는, 임시 저장된 공통 트랜잭션과 근심 트랜잭션 중 상기 합의된 트랜잭션을 하나의 임시 저장소에 저장하여 정렬을 진행하고, 그 결과값을 키노드에게 전송하여 검증받고, 키노드의 서명과 함께 결과값을 상기 복수 개의 노드들에게 전송하는 것을 특징으로 하며,

[0035] 상기 복수 개의 노드들은 상기 에이전트 노드으로부터 키노드의 서명과 함께 전송받은 결과값을 기반으로 하여 합의 과정없이 블록을 생성하는 것이 바람직하다.

[0036] 전술한 제1 특징에 따른 블록체인 네트워크에 있어서, 상기 블록체인 네트워크에 포함된 복수 개의 노드들의 각각은, 순차적으로, 범위 설정후, 블록의 해시값을 시드로 사용하여 랜덤 시드 알고리즘의 결과값을 설정하고, 상기 결과값을 이용하여 상기 복수 개의 노드들 중 하나의 키 노드를 선출하는 것이 바람직하다.

[0037] 본 발명의 제2 특징에 따른 블록체인 네트워크의 전처리 방법은, 복수 개의 노드들과 에이전트 노드를 구비하는 블록체인 네트워크에서의 전처리 방법으로서, 상기 복수 개의 노드들 중 하나의 키노드를 선출하는 단계; (b) 상기 키노드에 의해 에이전트 노드를 검증하고, 키노드의 검증이 성공적으로 완료되면 에이전트 노드에게 서명과 공개키를 전송하여 에이전트 노드를 활성화시키는 단계; (c) 상기 키노드의 상기 에이전트 노드에 대한 검증이 성공적으로 완료되면 에이전트가 활성화되었음을 상기 복수 개의 노드들에게 알리는 단계; (d) 상기 복수 개의 노드들이 상기 에이전트 노드에게 트랜잭션 풀(Transaction Pool)을 제공하는 단계; (e) 상기 에이전트 노드는 모든 노드들의 트랜잭션 풀을 검사하는 단계; (f) 상기 에이전트 노드는 모든 노드들의 트랜잭션 풀이 같으면, 키노드에게 검증받고 키노드의 서명과 함께 결과값을 상기 복수 개의 노드들에게 전송하는 단계; (g) 상기 복수 개의 노드들이 에이전트 노드로부터 키노드의 서명을 포함한 결과값을 기준으로 하여 합의 알고리즘없이 블록을 생성하는 단계;를 구비한다.

[0038] 전술한 제2 특징에 따른 에이전트 기반의 블록체인 네트워크에서의 전처리 방법은, (h) 상기 (e) 단계에서 모든 노드들의 트랜잭션 풀이 같지 않으면, 트랜잭션들을 분류하여 합의가 필요한 트랜잭션을 찾는 단계;를 더 구비하고,

[0039] 상기 (h) 단계는, 모든 노드가 동일하게 갖고 있는 트랜잭션은 공통 트랜잭션(Common Transaction)으로 분류하고, 공통 트랜잭션은 블록이 생성될 때까지 에이전트 노드가 임시 저장하고, 공통 트랜잭션 외의 트랜잭션은 근심 트랜잭션(Trouble transaction)으로 분류하는 것이 바람직하다.

[0040] 전술한 제2 특징에 따른 에이전트 기반의 블록체인 네트워크에서의 전처리 방법은, (i) 상기 근심 트랜잭션에 대하여 사전 설정된 합의 알고리즘을 적용하여 합의하고, 합의가 끝난 근심 트랜잭션은 합의된 트랜잭션으로 분류하는 단계; 를 더 구비하는 것이 바람직하다.

[0041] 전술한 제2 특징에 따른 에이전트 기반의 블록체인 네트워크에서의 전처리 방법은, (j) 상기 에이전트 노드는 상기 (h) 단계에서 임시 저장된 공통 트랜잭션과 상기 (i) 단계에서 상기 합의된 트랜잭션을 하나의 임시 저장소에 저장하여 정렬을 진행하고, 그 결과값을 키노드에게 전송하여 검증받고, 키노드의 서명과 함께 결과값을 상기 복수 개의 노드들에게 전송하는 단계;를 더 구비하는 것이 바람직하다.

[0042] 전술한 제2 특징에 따른 에이전트 기반의 블록체인 네트워크에서의 전처리 방법에 있어서, 상기 (a) 단계는, 블록체인 네트워크에 포함된 복수 개의 노드들에게, 범위 설정후, 블록의 해시값을 시드로 사용하여 랜덤 시드 알고리즘의 결과값을 설정하고, 상기 결과값을 이용하여 상기 복수 개의 노드들 중 하나의 키 노드를 선출하는 것이 바람직하다.

## *발명의 효과*

[0043] 전술한 본 발명에 따른 블록체인은 에이전트를 이용하여 전처리함으로써, 모든 노드들이 공통으로 갖고 있는 트랜잭션에 대하여는 합의 알고리즘의 적용없이 블록을 생성하게 된다. 또한, 모든 노드들의 트랜잭션들이 같지 않은 근심 트랜잭션에 대하여만 합의 알고리즘을 적용하여 합의하고, 합의된 근심 트랜잭션과 공통 트랜잭션들에 대하여 블록을 생성하게 된다.

[0044] 이와 같이 일부에 대하여 합의 알고리즘을 적용하지 않음으로써, 참여 노드가 급속하게 많아지더라도 합의에 따른 처리속도를 감소시키지 않으면서 블록체인 네트워크를 확장시킬 수 있게 된다.

## *도면의 간단한 설명*

[0045] 도 1은 본 발명의 바람직한 실시예에 따른 에이전트 기반의 블록체인 네트워크의 전처리 방법을 순차적으로 도시한 흐름도이다.

도 2는 본 발명의 바람직한 실시예에 따른 에이전트 기반의 블록체인 네트워크의 전처리 방법에 있어서, 키 노드 선출 의사 코드이다.

도 3은 본 발명의 바람직한 실시예에 따른 에이전트 기반의 블록체인 네트워크의 전처리 방법에 있어서, 에이전트 활성화 의사 코드이다.

도 4은 본 발명의 바람직한 실시예에 따른 에이전트 기반의 블록체인 네트워크의 전처리 방법에 있어서, 트랜잭션 풀 검사 의사 코드이다.

도 5는 본 발명의 바람직한 실시예에 따른 에이전트 기반의 블록체인 네트워크의 전처리 방법에 있어서, 트랜잭션 분류 및 합의 의사 코드이다.

도 6은 본 발명의 바람직한 실시예에 따른 에이전트 기반의 블록체인 네트워크의 전처리 방법에 있어서, 트랜잭션 정렬 의사 코드이다.

도 7은 악의적인 트랜잭션 비율에 따른 본 발명에 따른 알고리즘의 속도 변화를 알아보기 위한 실험 결과를 도시한 그래프이다.

도 8은 수학식 1을 이용하여 계산한 악의적인 트랜잭션 비율에 다른 기대 시간을 도시한 그래프이며, 도 9는 실험에 의해 측정된 악의적인 트랜잭션에 따른 처리시간을 도시한 그래프이며, 도 10은 수학식 1에 따라 예상되는 기대 시간 및 실험에 따라 측정된 처리 시간을 기반으로 하여 계산한 악의적인 트랜잭션에 따른 확장성을 도시한 그래프이다.

## *발명을 실시하기 위한 구체적인 내용*

[0046] 본 발명에 따른 블록체인 네트워크는 에이전트를 기반으로 하여 합의 알고리즘에 비잔틴 장애 허용을 적용할 수 있도록 하는 전처리 방법인 것을 특징으로 한다. 본 발명에 따른 에이전트는 합의의 확장성 개선을 도와주는 객체로 정의되며, 상기 에이전트는 특정한 간격으로 각 노드에 접근하여 전처리를 과정을 통해 합의가 필요한 순간을 알려주게 된다. 첫 번째 전처리 단계에서는 기록된 데이터가 모든 노드에서 일치하는지를 확인하고, 두 번째 전처리 단계에서는 어떤 데이터가 불일치하여 합의가 필요한지 알려준다. 프라이빗 블록체인 환경에서 실용적 비잔틴 장애 허용(PBFT) 합의 알고리즘에, 본 발명에 따라 에이전트를 기반으로 하여 전처리함으로써, 확장성이 종래의 방법들보다 개선된다.

[0047] 크리스마스 때 집에 방문하는 산타클로스처럼, 에이전트는 정의된 블록 생성 시간 간격으로 각 노드에 접근하여 합의가 필요한 순간을 알려주고 필요에 따라 합의를 진행하는 블록을 생성의 전반적인 과정을 돕는다. 에이전트는 블록체인 네트워크에서 분리된 독립적인 객체이며, 전처리 과정을 담당한다. 에이전트는 평상시에는 작동하지 않다가 선출된 임의의 노드(키노드)에 의해 작동을 시작하게 된다. 에이전트는 독립적인 객체이지만 블록체인에 영향을 줄 수 있는 의사결정은 혼자 내릴 수 없고 모든 처리 결과에 대해서 키노드의 서명을 받은 후 블록체인에 전파할 수 있다. 에이전트는 모든 데이터를 임시로 저장하고 전처리 및 합의 과정이 끝나는 대로 임시로 저장된 데이터를 삭제하여 데이터 유실 및 에이전트를 향한 해킹 위협을 방지한다.

[0048] 에이전트가 블록체인 네트워크에서 분리된 이유는 총 세 가지이다. 첫 번째로, 블록체인 내부의 특정 노드가 전처리를 담당하게 되면 노드가 전처리에 대한 부담을 안게 되므로 블록체인 내부 데이터 처리에 과부하를 줄 수 있다. 에이전트가 블록체인 네트워크에 포함되어 있으면, 에이전트의 악의적인 행동이 블록체인 네트워크에 직접적으로 영향을 끼칠 수 있다. 에이전트의 잘못된 행동으로 인해 블록체인 네트워크 노드들이 악의적인 내용을 수정하기 위한 자원이 낭비되어 블록체인의 확장성을 저하할 위험이 있다. 하지만 에이전트가 블록체인 네트워크로부터 분리되어 있어서 키노드를 통해 통제할 수 있으며, 만약 에이전트가 악의적인 행동을 하더라도 그 행동이 블록체인 네트워크에 영향을 주기 전에 사전 차단할 수 있다. 마지막으로 에이전트의 시스템 충돌이나 에이전트의 자체적인 문제가 발생하였을 때 블록체인 네트워크는 에이전트의 영향에서 벗어날 수 있고, 에이전트의 유지 보수가 쉬워진다는 장점이 있다.

[0049] 이하, 첨부된 도면을 참조하여 본 발명의 바람직한 실시예에 따른 에이전트 기반의 블록체인 네트워크의 전처리 방법에 대하여 구체적으로 설명한다.

[0050] 도 1은 본 발명의 바람직한 실시예에 따른 에이전트 기반의 블록체인 네트워크의 전처리 방법을 순차적으로 도시한 흐름도이다.

[0051] 도 1을 참조하면, 본 발명에 따른 에이전트 기반 전처리 기법은 합의 과정을 포함하여, 총 6단계로 구성된다. 첫 번째 단계(단계 100)에서는 에이전트를 작동시키기 위한 임의의 키노드를 선출하는 과정이다. 두 번째 단계(단계 110)는 작동하지 않고 대기하고 있는 에이전트를 키노드를 통해 작동시키기 위한 단계이다. 세 번째 및 네 번째 단계는 에이전트가 전처리를 진행하는 과정이다. 세 번째 단계(단계 120)에서 모든 노드가 같은 데이터를 가진 것이 확인된다면 네 번째 단계로 넘어 가지 않고 바로 블록을 생성한다(단계 160). 하지만 세 번째 단계에서 노드가 같은 데이터를 갖고 있지 않다면, 네 번째 단계(단계 130)로 넘어가서 트랜잭션 분류 작업을 진행하게 된다. 네 번째 단계 이후, 다섯 번째 단계(단계 140)는 이전 단계에서 분류된 합의가 필요한 트랜잭션을 합의하는 과정을 거친다. 마지막 여섯 번째 단계(단계 150)는 분류된 트랜잭션과 다섯 번째 단계(단계 140)에서 합의가 끝난 트랜잭션을 정렬하는 과정을 거치고, 블록을 생성하게 된다(단계 160). 이하, 전술한 각 단계에 대하여 보다 구체적으로 설명한다.

[0052] 먼저, 도 2를 참조하여 키노드 선출 단계(단계 100)에 대하여 구체적으로 설명한다. 도 2는 본 발명의 바람직한 실시예에 따른 에이전트 기반의 블록체인 네트워크의 전처리 방법에 있어서, 키 노드 선출 의사 코드이다.

[0053] 본 발명에서는 블록체인 노드들이 네트워크에 처음 참여할 때 고유한 식별 번호를 부여받게 되는 것을 가정한다. 도 2를 참조하면, 해당 알고리즘에서는 노드의 고유 식별번호를 1부터 네트워크에 접속한 순서대로 차례대로 정수를 부여하였다. 각각의 노드들은 블록 생성 시간 간격으로 랜덤 시드 알고리즘을 시작한다(line 4). 랜덤 시드 알고리즘은 범위를 설정하여 (line 5) 랜덤 시드 알고리즘의 결과 값이 고유 식별 번호 이외의 값이 나타나지 않도록 설정하였다. 범위 설정 뒤, 블록의 해시값을 시드로 사용하여 랜덤 시드 알고리즘(Random Seed Algorithm)을 통해 임의의 노드를 선출한다(line 6). 이전 블록의 해시값은 이미 모든 노드가 합의한 것을 의미하므로 모든 노드는 같은 임의의 숫자를 획득할 수 있다. 획득된 임의의 숫자를 통해 특정 라운드의 키노드

가 식별된다(line 7). 키노드에 선출된 노드는 본인의 고유 식별 번호와 이전 블록의 해시값을 개인 키를 암호화하여 서명을 만들게 되고(line 8) 본인의 개인 키로 만든 공개키를 함께 묶어서 에이전트에게 전송한다(line 9). 키노드의 서명을 노드의 고유 번호와 이전 블록의 해시값으로 만들게 함으로써, 키노드가 선출 될 때마다 매번 다른 서명을 생성하여 악의적인 노드가 키노드처럼 행동하는 것을 방지한다.

[0054] 다음, 도 3을 참조하여 키노드를 통한 에이전트 활성화 단계(단계 110)에 대하여 보다 구체적으로 설명한다. 도 3은 본 발명의 바람직한 실시예에 따른 에이전트 기반의 블록체인 네트워크의 전처리 방법에 있어서, 에이전트 활성화 의사 코드이다.

[0055] 도 3을 참조하면, 에이전트는 스스로 작동하지 않으며, 키노드로부터 서명을 받기 전까지 에이전트는 작동하지 않는다. 에이전트는 키노드로부터 서명과 공개키를 전송받아서 키노드가 전송한 서명과 키노드의 공개키가 맞는지 검증을 진행한다(line 1). 랜덤 시드 알고리즘의 결과 값이 키노드의 고유 식별 번호와 다르거나 전송받은 서명과 공개키가 랜덤 시드 알고리즘으로 선출된 키노드가 전송한 것이 아니라면 에이전트는 제대로 된 키노드의 서명과 공개키가 전송될 때까지 작동하지 않는다(line 6). 키노드의 검증이 성공적으로 완료된 경우 에이전트는 작동되었음을 모든 노드에 알린 뒤(line 3) 전처리 과정을 진행한다(line 4).

[0056] 다음, 도 4을 참조하여 트랜잭션 풀 검사 단계(단계 120)에 대하여 보다 구체적으로 설명한다. 도 4은 본 발명의 바람직한 실시예에 따른 에이전트 기반의 블록체인 네트워크의 전처리 방법에 있어서, 트랜잭션 풀 검사 의사 코드이다.

[0057] 첫 번째 전처리 과정은 합의가 필요 없는 상황이라도 참여자의 확인을 받기 위해 진행되던 기존의 합의 과정을 생략하기 위함이다. 에이전트가 작동되었다는 메시지를 받은 블록체인 노드들은 스스로 검증한 트랜잭션 및 과거에 기록된 모든 데이터를 저장한 저장소인 트랜잭션 풀(Transaction Pool)을 에이전트에게 공개한다. 도 4를 참조하면, 에이전트는 각각의 노드에 접근하여 특정 기준에 의해 정렬된 트랜잭션 풀을 가져오면서(line 1) 해시함수를 사용하여 트랜잭션 풀을 해시 형태로 변환한다(line 2). 해시 함수로 변환된 트랜잭션 풀은 에이전트의 임시 저장소에 저장된다(line 3). 해시 함수의 특성상 네트워크에 참여하고 있는 노드가 같은 데이터를 트랜잭션 풀에 저장하였다면 에이전트 임시 저장소에 모두 같은 해시값이 나타날 것이고, 어떠한 노드가 하나라도 다른 데이터를 트랜잭션 풀에 저장하였다면 에이전트 임시 저장소에 다른 해시값이 표현될 것이다. 에이전트는 모든 노드가 같은 트랜잭션 풀 해시값을 가지고 있는 것을 확인 후(line 4), 키노드에게 전처리 결과를 검증받는다(line 5). 키노드의 검증이 성공적으로 이루어지면, 에이전트는 키노드의 서명과 함께 결과 값을 다른 노드들에 전송함으로써(line 7) 노드들은 전달받은 결과값을 기준으로 합의 알고리즘을 사용할 필요 없이 스스로 블록을 생성한다. 만약 노드 중 하나라도 다른 트랜잭션 풀 해시값을 갖는다면 에이전트는 두 번째 전처리 과정을 진행한다(line 10).

[0058] 다음, 도 5를 참조하여 합의가 필요한 트랜잭션 식별 단계(단계 130)에 대하여 보다 구체적으로 설명한다. 도 5는 본 발명의 바람직한 실시예에 따른 에이전트 기반의 블록체인 네트워크의 전처리 방법에 있어서, 트랜잭션 식별 및 분류에 대한 의사 코드이다. 두 번째 전처리 과정은 이전 전처리 과정에서 노드들이 같은 데이터를 가지고 있지 않으면 합의가 필요한 대상을 찾기 위한 트랜잭션 분류 작업이다.

[0059] 도 5를 참조하면, 에이전트는 전술한 트랜잭션 풀 검사에서 기존 노드로부터 가져온 트랜잭션 풀을 활용하여 트랜잭션들을 하나하나 분류한다. 모든 노드가 동일하게 가지고 있는 트랜잭션이라면 공통 트랜잭션(Common Transaction)으로 분류(line 2)하고, 공통 트랜잭션은 블록이 생성될 때까지 에이전트가 임시로 저장을 한다(line 3). 그 외의 트랜잭션을 근심 트랜잭션 (Trouble Transaction)으로 분류한다(line 5).

[0060] 다음, 근심 트랜잭션 합의 단계(단계 140)에 대하여 보다 구체적으로 설명한다. 상기 근심 트랜잭션 합의 과정은 합의가 필요한 트랜잭션 식별 단계에서 분류된 근심 트랜잭션만을 가지고 합의를 진행한다. 이 과정에서는 어떠한 합의 알고리즘도 적용될 수 있으며, 합의가 끝난 트랜잭션은 합의된 트랜잭션(Consensus Transaction)으로 분류를 한다. 이 과정에서 합의가 되지 않은 근심 트랜잭션은 완전히 제외된다.

[0061] 다음, 도 6을 참조하여 트랜잭션 정렬 단계(단계 150)에 대하여 보다 구체적으로 설명한다. 도 6은 본 발명의 바람직한 실시예에 따른 에이전트 기반의 블록체인 네트워크의 전처리 방법에 있어서, 트랜잭션 정렬 의사 코드이다. 상기 트랜잭션 정렬 과정은 이전에 이루어졌던 모든 과정을 정리하여 노드들에 전파하기 위한 과정이다. 도 6을 참조하면, 에이전트는 합의가 필요한 트랜잭션 식별 과정에서 분류된 공통 트랜잭션과 근심 트랜잭션의 합의 과정에서 합의가 끝난 합의된 트랜잭션을 하나의 임시 저장소에 저장하여 정렬을 진행한다(line 1). 이 과정은 블록체인에서 제공하는 고유의 정렬 기법을 활용할 수 있다. 본 발명에서는 트랜잭션들의 타임 스탬프기준

으로 정렬하고(line 2), 결과 값을 키노드에게 전송한다(line 3). 키노드는 에이전트에게 전달받은 값을 확인 후 서명을 하고, 서명된 값을 받은 에이전트는 블록체인 노드들에 전송한다(line 4).

[0062] 에이전트로부터 키노드에 의해 서명된 값을 받은 전송받은 블록체인 네트워크의 노드들은 블록을 생성하게 된다.

[0063] 본 발명에 따른 방법을 이용하여 합의에 걸리는 평균 기대 시간을 수학식 1로 표현할 수 있다.

*수학식 1*

[0064] $$Expected\ Time = P_1 + f(P_2 + Cs)$$

[0065] 평균 기대 시간이란 본 발명에 따른 알고리즘이 하나의 블록을 생성할 때까지의 과정을 무한 실행했을 때의 소요되는 평균처리 시간을 의미한다. P1은 첫 번째 전처리 과정인 트랜잭션 풀 검사의 처리 시간을 의미하며, P2는 두 번째 전처리 과정인 합의가 필요한 트랜잭션 식별의 처리 시간을 의미한다. Cs는 근심 트랜잭션 합의의 처리 시간을 의미하며, f는 악의적인 트랜잭션의 비율을 의미한다. 악의적인 트랜잭션이 있는 환경이라도 P1은 노드가 같은 데이터를 가졌는지 확인하기 위해서 반드시 실행되고, P2와 Cs는 확률적으로 적용이 되기 때문에 위와 같은 수학식 1이 완성되었다. 수식에 의하면 악의적인 트랜잭션 비율이 적정 수준이 되기 전까지는 본 발명에 따른 알고리즘을 적용하는 것이 그렇지 않을 때와 비교하여 더 빠를 것으로 기대할 수 있다.

[0066] 이하, 본 발명에 따른 블록체인 네트워크의 성능을 확인하기 위한 실험을 진행한 결과를 설명한다. 본 실험에서는 본 발명에 따른 알고리즘의 적용을 통해 처리 속도가 어느 정도 빨라지는지를 실험하며, 가정한 수학식 1로 계산되는 결과와 실험 사이에서 차이를 보이는지 실험한다. 이와 더불어, 본 발명에 따른 알고리즘 확장성의 한계를 알아보기 위한 실험을 진행한다. 실험을 진행하기 위해서 프라이빗 블록체인에서 가장 많이 활용되고 있는 BFT 기반의 알고리즘인 PBFT 합의 과정을 사용하였다. PBFT 알고리즘의 경우 노드 수가 증가함에 따라 처리 속도가 현격히 늘어나게 되므로, 본 발명에 따른 알고리즘을 적용할 때와 적용하지 않을 때를 비교하여 어느 정도 처리 시간이 빨라지는지를 확인할 수 있다. 노드의 수와 악의적 트랜잭션의 비율을 조절하며 매 1,000회의 합의를 통해 평균 처리 시간을 계산한다. 그리고 같은 조건에서 산타클로스를 적용한 PBFT와 적용하지 않은 PBFT에서 평균 처리 시간을 비교한다. 임의의 트랜잭션을 10ms 간격으로 발생시켰으며 블록은 매 10초 간격으로 생성하는 것으로 가정하였다. 설정된 악의적 트랜잭션의 비율에 맞게 임의로 악의적 트랜잭션을 발생시켜 특정 비율로 근심 트랜잭션 합의 과정에 진입하게 유도하였다. 노드 및 에이전트의 하드웨어 성능은 CPU: Intel i5-3570, 3.4GHz, RAM: 4GB, OS: Windows10 이다.

[0067] 도 7은 악의적인 트랜잭션 비율에 따른 본 발명에 따른 알고리즘의 속도 변화를 알아보기 위한 실험 결과를 도시한 그래프이다. 노드 수는 4개부터 100개까지 4개씩 증가시켰다. PBFT 알고리즘은 악의적인 트랜잭션이 없는 환경에서 측정하였다. 본 발명에 따른 알고리즘과 PBFT의 조금 더 정확한 처리속도 비교를 위해 PBFT 알고리즘이 허용하는 최대 1/3의 장애 허용률을 고려하여 악의적인 트랜잭션이 33%일 때의 본 발명에 따른 알고리즘의 처리 속도 또한 측정하였다. 전체적으로 노드가 증가함에 따라서 처리해야 하는 트랜잭션의 양이 늘어나기 때문에 처리 시간 또한 점차 증가하는 것을 확인할 수 있다.

[0068] 도 7을 참조하면, 악의적인 트랜잭션이 0% 일 때는 속도 변화가 거의 없는 것처럼 확인이 되나, 실질적으로는 노드 수가 증가함에 따라서 검사해야 하는 트랜잭션 풀 해시값이 많아짐으로써 처리속도가 조금씩 증가한다. 악의적인 트랜잭션이 하나도 없을 때는 트랜잭션 풀 검사 과정만을 거치기 때문에 처리 속도가 도 7에서 가장 빠르게 나타났다. 또한, 전체 트랜잭션 풀에서 악의적인 트랜잭션의 수가 33%가 된다고 하더라도 악의적인 노드가 없는 PBFT 알고리즘 보다 약 39% 더 빠른 처리 속도를 가지고 있는 것을 실험을 통해서 확인할 수 있다. 기존 PBFT에서는 올바른 트랜잭션을 다른 노드들에 확인받기 위해 계속 합의를 진행하였으나, 본 발명에 따른 알고리즘은 합의가 필요한 트랜잭션만을 분류하여 올바른 트랜잭션은 제외하고 합의가 필요한 트랜잭션만을 가지고 합의를 진행하였기에 악의적인 트랜잭션이 33%가 있다고 하더라도 기존의 PBFT보다 빠른 처리 속도를 보인다.

[0069] 도 8은 수학식 1을 이용하여 계산한 악의적인 트랜잭션 비율에 다른 기대 시간을 도시한 그래프이며, 도 9는 실험에 의해 측정된 악의적인 트랜잭션에 따른 처리시간을 도시한 그래프이며, 도 10은 수학식 1에 따라 예상되는 기대 시간 및 실험에 따라 측정된 처리 시간을 기반으로 하여 계산한 악의적인 트랜잭션에 따른 확장성을 도시

한 그래프이다.

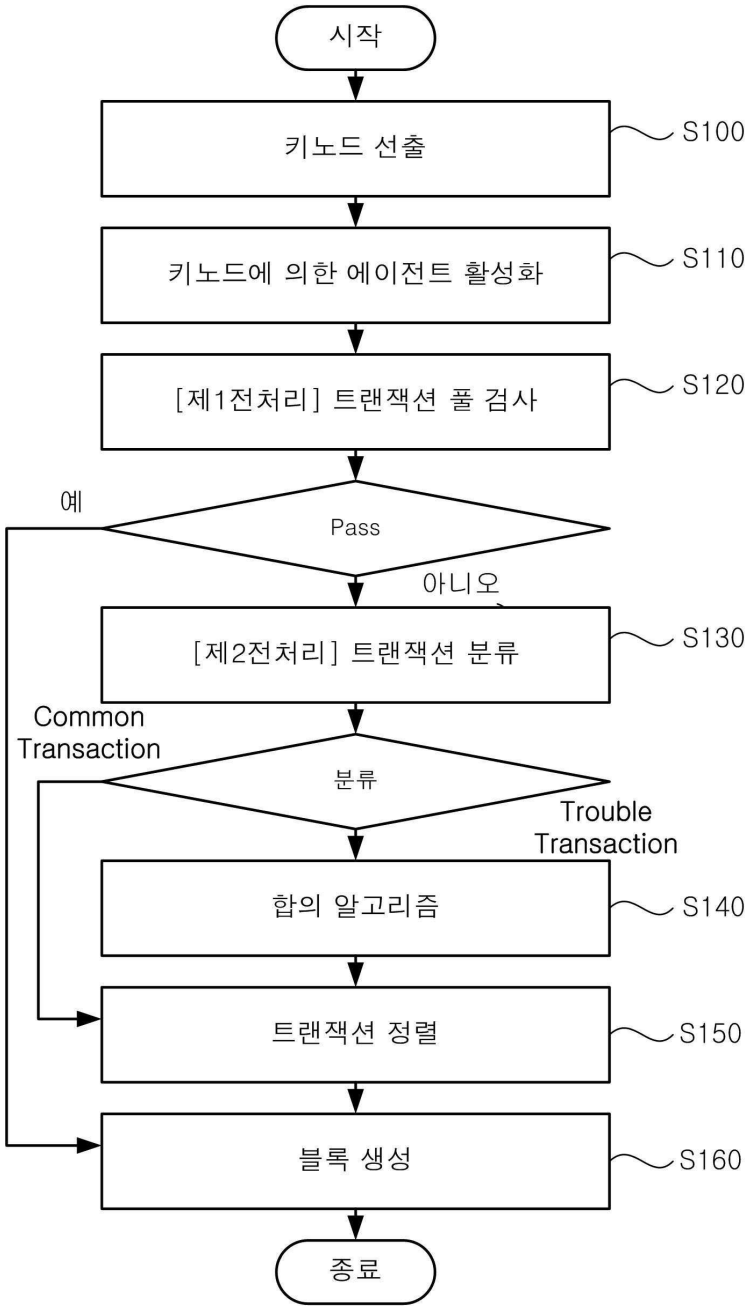
[0070] 상기 도 8, 9, 10의 실험은 PBFT 알고리즘을 사용하였을 때의 본 발명에 따른 알고리즘의 확장성 한계를 알아보기 위한 실험이다. 본 실험은 PBFT를 사용했을 때의 본 발명에 따른 알고리즘의 확장성의 한계를 파악함으로써, 어떤 상황에 본 발명에 따른 알고리즘을 적용시키는 것이 순수 PBFT를 사용하는 것보다 빠른지 여부를 파악하기 위한 실험이다.

[0071] 도 10의 경우, 도 8 및 도 9에서 각 노드 수에 따른 본 발명에 따른 알고리즘과 PBFT 알고리즘의 처리 속도가 만나는 지점(Interception point)을 연결하여 나타낸 그래프이다. 그래프의 위쪽 범위에 포함된 상황에서는 PBFT 알고리즘이 본 발명에 따른 알고리즘보다 처리 속도가 더 빠르다는 것을 의미한다. 예를 들어서 노드 수가 52개인 환경에서 악의적인 트랜잭션의 비율이 약 84% 이상이라면 본 발명에 따른 알고리즘을 제외한PBFT 알고리즘만을 사용하는 것이 처리속도가 더 빠르다는 것을 의미한다. 반대로 그래프의 아래쪽 범위는 본 발명에 따른 알고리즘을 사용한 처리 속도가 PBFT 알고리즘보다 빠른 환경을 의미한다. 도 10을 통해 본 발명에 따른 알고리즘을 적용한 PBFT의 확장성이 순수한 PBFT보다 더 크다는 것이 확인할 수 있다. 또한, 본 발명에 따른 알고리즘은 PBFT 합 의 알고리즘을 적용했을 때 네트워크의 노드 수가 적을 때는 약 68~70%의 악의적인 트랜잭션 비율을 허용하며, 노드 수가 많은 환경에서는 약 80~85%의 악의적인 트랜잭션을 허용한다는 것을 확인 할 수 있다.

[0072] 이상에서 본 발명에 대하여 그 바람직한 실시예를 중심으로 설명하였으나, 이는 단지 예시일 뿐 본 발명을 한정하는 것이 아니며, 본 발명이 속하는 분야의 통상의 지식을 가진 자라면 본 발명의 본질적인 특성을 벗어나지 않는 범위에서 이상에 예시되지 않은 여러 가지의 변형과 응용이 가능함을 알 수 있을 것이다. 그리고, 이러한 변형과 응용에 관계된 차이점들은 첨부된 청구 범위에서 규정하는 본 발명의 범위에 포함되는 것으로 해석되어야 할 것이다.

도면

도면1


시작
키노드 선출
S100
키노드에 의한 에이전트 활성화
S110
[제1전처리] 트랜잭션 풀 검사
S120
예
Pass
아니오
[제2전처리] 트랜잭션 분류
S130
Common Transaction
분류
Trouble Transaction
합의 알고리즘
S140
트랜잭션 정렬
S150
블록 생성
S160
종료

도면2

**Algorithm 1** Key Node Election

$Max_{range}$ = number of blockchain node
$t$ = block generation interval
$Seed$ = previous block hash
**if** Timer = $t$ **then**
    $Set_{range} = 1$ *to* $Max_{range}$
    $Result_{random}$ = random algorithm ($Set_{range}$, $Seed$)
    **if** $Node_{idNumber} = Result_{Random}$ **then**
        $Key_{sig} = Key_{pri}$ ($Node_{idNumber}$, $Seed$)
        send to Agent ($Key_{sig}$, $Key_{pub}$)
    **end if**
**end if**

도면3

**Algorithm 2** Agent Activation

**if** $Key_{sig}, Key_{pub} \in Key_{node}$ **then**
    $Agent_{state}$ = ON
    broadcast ($Agent_{state}$)
    *first Preprocessing* ($TxPool_n$)
**else**
    $Agent_{state}$ = OFF
**end if**

도면4

**Algorithm 3** First Preprocessing

access $TxPool_n$
= {$transaction_1$, $transaction_2$, $\cdots$ $transaction_i$} $\leftarrow TxPool_n \in Node_n$
hash algorithm ($TxPool_n$)
add $Hash_{txPooln}$ *to* $Hash_{result}$ = {$Hash_{txPool1}$, $Hash_{txPool2}$, $\cdots$ $Hash_{txPooln}$}
**if** $Hash_{result}$ elements are same **then**
    send to $Key_{node}$ ($Result_{p1}$)
    **if** $Result_{p1}$ = valid **then**
        broadcast ($Result_{p1}$)
    **end if**
**else**
    *Second Preprocessing* ($TxPool_n$)
**end if**

도면5

**Algorithm 4** Second Preprocessing

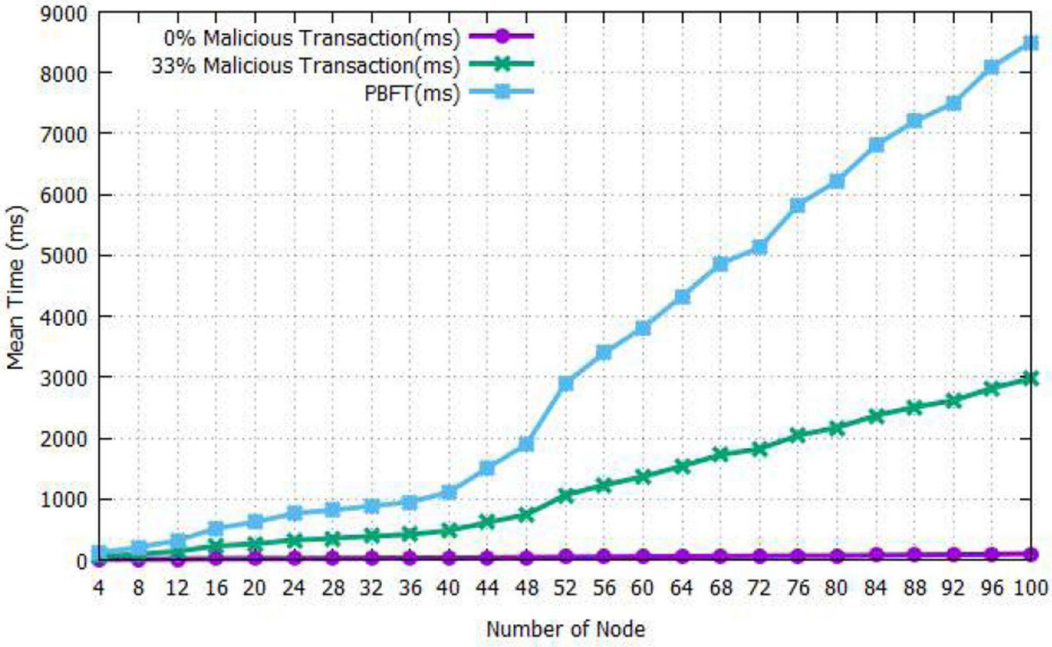
1: **if** $transaction_i = Common_{transaction}$ **then**
2: add to $C_t = \{Common_{transaction1}, Common_{transaction2}, \cdots Common_{transactioni}\}$
3: save $C_t$ until block generate
4: **else**
5: add to $T_t = \{Trouble_{transaction1}, Trouble_{transaction2}, \cdots Trouble_{transactioni}\}$
6: execute consensus algorithm with $T_t$
7: **end if**

도면6

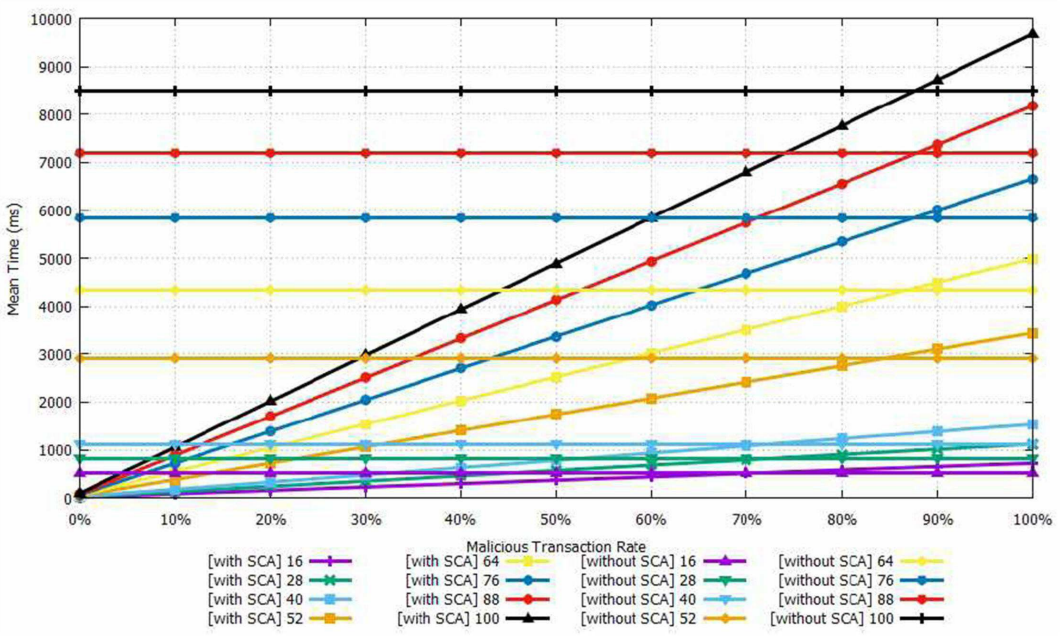
**Algorithm 5** Sorting

1: $Temp_Result_{p2} = \{C_t \cup Consensus_t\}$
2: $Result_{p2}$ = sort $(Temp_Result_{p2})$
3: send to $Key_{node}(Result_{p2})$
4: **if** $Result_{p2}$ = valid **then**
5: broadcast $(Result_{p2})$
6: **end if**

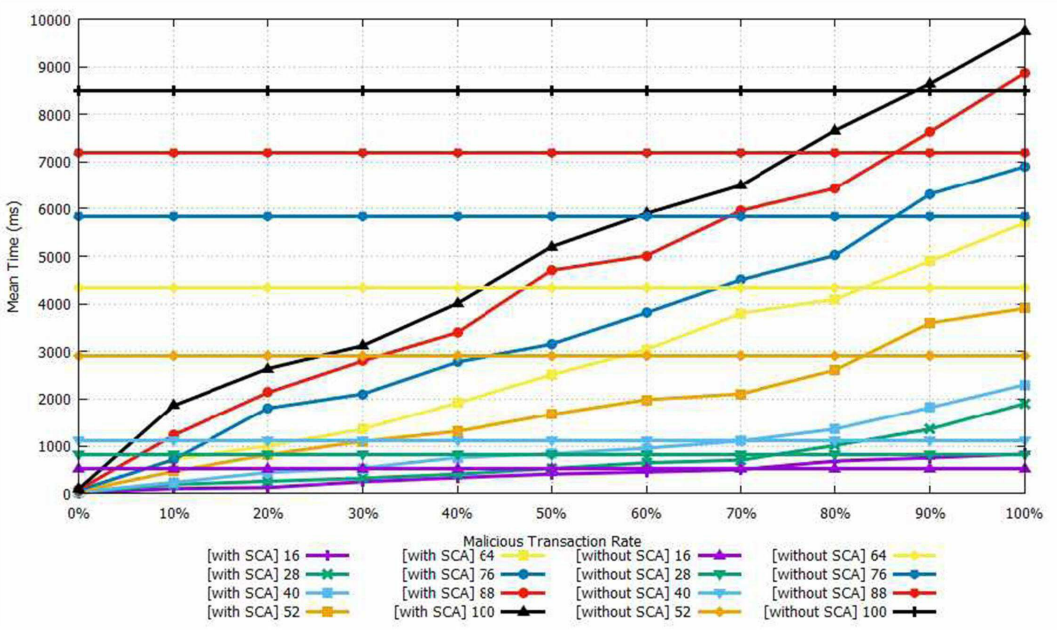
도면7



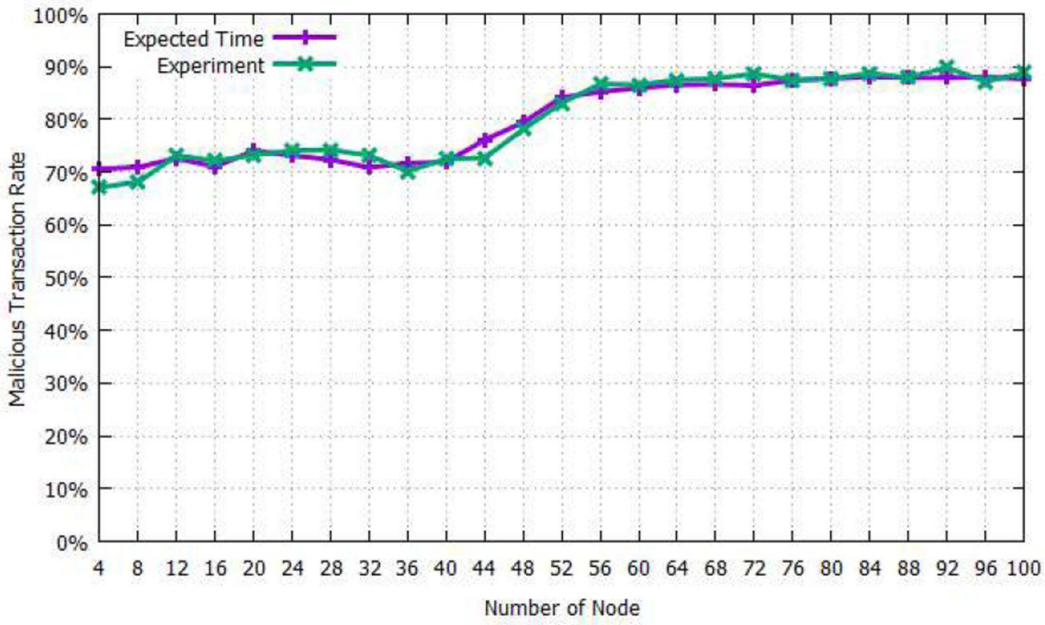
도면8


Mean Time (ms)
Malicious Transaction Rate
[with SCA] 16
[with SCA] 28
[with SCA] 40
[with SCA] 52
[with SCA] 64
[with SCA] 76
[with SCA] 88
[with SCA] 100
[without SCA] 16
[without SCA] 28
[without SCA] 40
[without SCA] 52
[without SCA] 64
[without SCA] 76
[without SCA] 88
[without SCA] 100

도면9


Mean Time (ms)
Malicious Transaction Rate
[with SCA] 16
[with SCA] 28
[with SCA] 40
[with SCA] 52
[with SCA] 64
[with SCA] 76
[with SCA] 88
[with SCA] 100
[without SCA] 16
[without SCA] 28
[without SCA] 40
[without SCA] 52
[without SCA] 64
[without SCA] 76
[without SCA] 88
[without SCA] 100

도면10


Expected Time
Experiment
Malicious Transaction Rate
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
4 8 12 16 20 24 28 32 36 40 44 48 52 56 60 64 68 72 76 80 84 88 92 96 100
Number of Node