연합학습을 위한 이더리움 스마트 컨트랙트 자료구조 효율성 분석

김민석 [1] 박상현 [2] 문수묵 [2]
[1] 서울대학교 경영학과
[2] 서울대학교 전기정보공학부
seoultower04@snu.ac.kr, lukepark@snu.ac.kr, smoon@snu.ac.kr

Efficient Ethereum Smart Contract Data Structure for Federated Learning

Minseok Kim[1] Sanghyeon Park [2] Soo-Mook Moon [2]
[1] Department of Business Administration, Seoul National University
[2] Department of Electrical and Computer Engineering, Seoul National University

요 약

분산시스템에서는 중앙시스템의 통제없이 시스템을 관리하는 합의방식이 필요하다. 분산 딥러닝인 연합학습에서 중앙서버의 역할까지 분산시스템이 대체하는 경우, 합의를 위해 블록체인이 사용될 수 있다. 본 연구는 블록체인의 스마트 컨트랙트를 통해 연합학습을 제어하는 경우, 스마트 컨트랙트가 가질 수 있는 2 개의 자료구조를 소개하며, 각 자료구조에 따른 이더리움 블록체인상의 가스비를 분석함을 통해 경제적 효율화를 시도한다.

1. 서 론

분산시스템에서는 시스템을 통제하는 중앙주체 없이도 시스템을 관리하는 민주적 합의방식[1]을 마련하는 것이 중요하다. 따라서, 분산시스템에 대한 수요가 증가할수록, 더욱 안정적이고 효율적인 합의의 방식을 고안할 필요가 있다.

본 연구에서는 분산 딥러닝인 연합학습(Federated Learning) 시스템[2]의 완전한 탈중앙화 과정에서 시스템의 합의를 위해 블록체인 스마트 컨트랙트(Smart Contract)를 사용하는 경우에 스마트 컨트랙트가 가지는 자료구조와 그에 따른 경제성을 분석한다. 스마트 컨트랙트는 블록체인 상에 배포되고 어느 노드에도 소속되지 않은 독립된 객체로서 노드별 신뢰도를 집계 및 업데이트 한다. 이를 통해 연합학습 분산시스템은 중앙주체 없이도 서로의 신뢰도를 확인하고 민주적인 평가 및 통제를 할 수 있다. 따라서 이러한 시스템의 기반이 되는 스마트 컨트랙트의 자료구조를 최적화하는 연구가 요구된다.

이 논문은 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. 2020R1A2B5B02001845).

본 연구의 흐름은 다음과 같이 구성된다. 2 장에서는 연구의 기반이 되는 블록체인과 스마트 컨트랙트 기술을 간략히 소개한다. 3 장에서는 연합학습을 위한 Queue 와 medianHeap 기반 스마트 컨트랙트의 자료구조를 소개하고 스마트 컨트랙트 내 upload 알고리즘의 시간복잡도를 측정한다. 마지막으로, 4 장에서는 두 자료구조별 연산량을 계측하여 경제성을 분석한다.

2. 배경지식

2.1 블록체인

블록체인은 분산 시스템 기술과 암호학을 기반으로 하는 분산 원장 기술이다. 합의알고리즘을 기반으로 누구나 자유롭게 참여하고 동일한 상태(state)를 공유하게 한다. 또한, 누구나 상태 전이를 제안할 수 있다. 참여자는 시스템에 참여하기 위해 주소값을 가지며, 상태 전이를 위해 트랜잭션(transaction)을 생성 및 P2P 네트워크상에 전파하면, 채굴자 노드가 이를 수집해 블록을 생성한다.

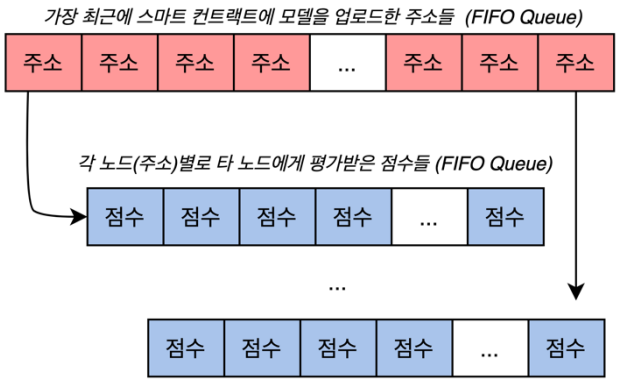

그림 1 Queue 자료구조 기반 스마트 컨트랙트

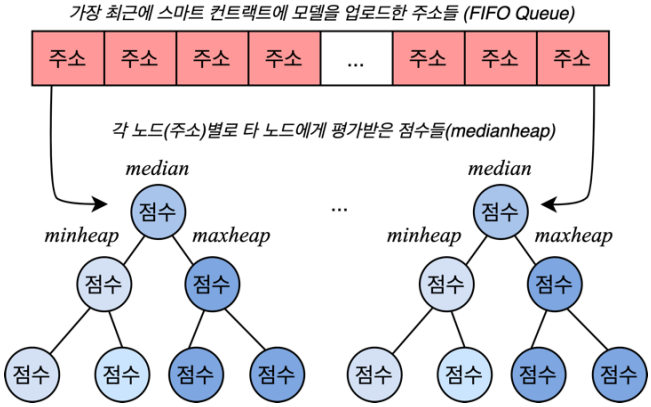

그림 2 MedianHeap 자료구조 기반 스마트 컨트랙트

전체 시스템에서는 작업증명(Proof-of-Work) 등 다양한 합의 알고리즘을 통해 블록을 검증 및 확정하고, 블록체인에 기록한다[3]. 즉, 블록체인은 중앙주체의 통제없이 분산시스템이 동일한 상태를 공유하고, 업데이트 할 수 있도록 하는 기술이다.

2.2 스마트 컨트랙트

스마트 컨트랙트(Smart Contract)는 이더리움(Ethereum)을 시작으로 블록체인에 도입된 개념으로, 미리 정의된 특정 조건이 만족될 때 특정 작업을 자동적으로 실행시키는 역할을 한다[4].

스마트 컨트랙트는 누구나 생성할 수 있으며, 생성 후 배포되면 고유 주소를 가지고 존재한다. 이러한 주소에 트랜잭션을 보내는 것으로 상태를 업데이트 할 수 있다.

트랜잭션은 연산량에 비례하여 비용(fee)이 발생한다. 이더리움에는 연산 단위로 가스(gas)를 사용하며, 단위 가스 당 소요되는 비용을 가스 가격(gas price)이라 한다.한 트랜잭션에서 사용된 가스의 총량을 사용 가스(gas used)라 할 때, 비용(gas cost)은 다음과 같이 계산된다.

$$gas\ cost = gas\ used * gas\ price$$

본 연구에서는 이더리움 스마트 컨트랙트의 연산 복잡도(complexity)를 분석하기 위한 척도로 트랜잭션의 연산량에 따른 비용인 가스 비용을 사용한다.

3. 연합학습을 위한 컨트랙트 자료구조와 알고리즘

본 연구에서는 스마트 컨트랙트에 기반한 연합학습을 효율적으로 제어하는 컨트랙트 구조를 고안하였다. 스마트 컨트랙트는 노드별 상호 평가 신뢰도를 집계 및 업데이트해 보관하며, 특정 노드가 정보를 요청하면 보관하고 있던 타 노드들에 관한 정보를 전달한다.

Algorithm 1: Upload() algorithm in smart contract

A_r : Queue containing the addresses with the most recent uploads
P_a : Data structure containing addresses and corresponding points of each addresses in A_r
T_a : teacher's addresses
t_a : each teacher address in T_a
T_p : teacher's points
t_p : each teacher point in T_p
Input : (T_a, T_p)
for *each teacher address* t_a *in* T_a **do**
  if t_a *exists in* A_r *and* P_a **then**
    delete t_a and t_p in P_a
  end
  insert t_a and t_p in P_a
end
if *the function caller is not in* A_r **then**
  if A_r *is full* **then**
    pop last element of A_r
  end
  insert function caller's address in A_r
end

알고리즘 1 함수 upload 의 수도코드

3.1 Queue 기반 스마트 컨트랙트

큐(Queue)기반 스마트 컨트랙트는 그림 1 처럼 두개의 큐로 구성된다. 하나는 블록체인 스마트 컨트랙트 상에 가장 최근에 모델을 업로드한 노드들의 주소를 보관하는 큐이다. 다른 하나는 각 노드별로 타 노드들에게 평가받은 점수들을 보관하는 큐이다.

또한, 스마트 컨트랙트는 각 노드별로 학습을 위해 배정해줄 타 노드들의 주소를 가지고 있다. 컨트랙트는 노드가 요청하면 보관정보들을 전달한다.

3.2. MedianHeap 기반 스마트 컨트랙트

중위힙(Median Heap)기반 스마트 컨트랙트는 그림 2 처럼 큐와 중위힙 구조를 통하여 구성된다. 큐는 스마트 컨트랙트상에 가장 최근에 모델을 업로드한 노드들의 주소값을 보관하는 큐이다.

중위힙은 각 노드별로 타 노드에게 평가받은 점수를 중위정렬하여 보관하는 힙(heap) 구조이다. 마찬가지로 컨트랙트는 노드의 요청에 따라정보를 전달해준다.

비용 (Gas)	medianheap		queue	
	거래	실행	거래	실행
배포	1936307	1420395	1088625	800109
upload	262512	237806	191037	172731

표 1 자료구조에 따른 스마트 컨트랙트 비용 분석결과

3.3 *UPLOAD* algorithm

임의의 노드는 트랜잭션을 통해 스마트 컨트랙트 내에 정의된 함수를 호출(function call) 및 실행한다. 연합학습을 위한 스마트 컨트랙트 구조에서는 가장 핵심이 되는 것이 다음의 upload 함수이기 때문에, 본 연구에서는 upload 의 실행비용을 분석했다.

함수 upload 는 한 노드가 주변의 여러 노드에 대해 평가한 결과를 컨트랙트에 업로드한다. 알고리즘 1 은 upload 의 동작을 수도코드(pseudo-code)로 나타낸 것이다.

주변 노드를 저장한 배열의 길이를 *N* 이라 하자. 반복문에서는 각 주변 노드들의 주소에 대하여, 점수를 저장하는 큐 또는 중위힙 구조에 새 점수를 삽입(insert)하는데, 어느 경우나 삽입의 복잡도는 *O(1)* 이므로 반복문의 총 시간복잡도(time complexity)는 *O(N)* 이다. 또한, 추출(pop) 과 삽입은 큐 자료구조에서 이루어지는 복잡도가 *O(1)* 인 작업이다. 종합하자면, 알고리즘의 전체 시간복잡도는 *O(N)* 이다.

4. 자료구조별 경제성 분석 및 최적화

스마트 컨트랙트의 특성상, 자료구조에 따른 upload 함수의 시간복잡도가 동일하더라도, 실제 실행시간 상의 미세한 차이와 공간복잡도(space complexity)의 차이 등을 고려할 필요가 있다. 즉, 두 자료구조 사이의 경제성 차이를 분석하고, 최적 자료구조를 찾는 과정이 요구된다.

이를 위한 실험 환경은 다음과 같다. 이더리움 스마트 컨트랙트 개발을 위한 웹 개발도구인 리믹스(remix)를 활용했으며, 이더리움 가상머신을 모방하는 Javascript VM 을 활용했다. 스마트 컨트랙트 구현에는 솔리디티(solidity) 언어를 활용했다.

두 자료구조에 따른 스마트 컨트랙트 배포(deploy), 거래(transaction) 및 실행(execution)비용 분석 결과는 표 1 과 같다. 두 자료구조에는 동일한 두 개의 주변 노드 주소와 점수를 인자로 주었으며, 각각 5 번의 거래(transaction)에서 발생한 가스 비용의 평균값을 기록한 것이다.

실험결과에 따르면, 배포, 거래, 실행의 모든 측면에서 큐 기반 스마트 컨트랙트의 경제성이 더욱 크다는 것을 확인할 수 있다. 이는 스마트 컨트랙트를 활용한 분산시스템의 경제성을 극대화하기 위해서 컨트랙트 구현에 중위힙 보다는 큐 구조를 활용해야 한다는 점을 시사한다.

5. 결론 및 향후 연구

본 연구에서는 중앙 주체가 없는 연합학습 시스템을 통제하는 블록체인 스마트 컨트랙트 자료구조의 경제성을 분석했다. 스마트 컨트랙트가 연합학습 노드별로의 상호신뢰도를 집계하는 큐와 중위힙 기반 자료구조를 구현하였으며, 이더리움 블록체인 환경 상에서 실험을 진행하였다. 실험 결과, 스마트 컨트랙트에서 연산량의 척도인 가스 사용량을 기준으로 했을 때, 스마트 컨트랙트 구현을 위해 힙보다는 큐 구조를 사용해야 함을 알 수 있었다.

본 연구에서는 분산시스템의 조정을 가능케 하는 두 자료구조로 큐와 힙을 비교했으나, 향후 연구에서는 더욱 다양한 형태의 자료구조에 대한 성능 및 가스 사용량을 비교할 예정이다. 또한 시간복잡도를 분석할 때, 각각의 자료구조가 가득차지 않은(not full) 상태만을 고려해 시간상환분석(amortized time complexity)만을 진행했으나, 향후 연구에서는 알고리즘 별로 발생할 수 있는 특이 경우를 추가적으로 반영할 예정이다.

참 고 문 헌

[1] Ongaro, Diego, and John Ousterhout. "In search of an understandable consensus algorithm." In *2014 {USENIX} Annual Technical Conference ({USENIX}{ATC} 14)*, pp. 305-319. 2014.

[2] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, Blaise Agueray Arcas, "Communication - Efficient Learning of Deep Networks from Decentral-ized Data," inProceedings of the 20th International Conference on Artificial In-telligence and Statistics, PMLR 54:1273-1282, 2017.0

[3] Nakamoto, Satoshi. *Bitcoin: A peer-to-peer electronic cash system*. Manubot, 2019.

[4] Buterin, Vitalik. "A next-generation smart contract and decentralized application platform." *white paper* 3, no. 37 (2014).