Backend Developer

PORTFOLIO

kang seungsu

CONTENTS

Part 1,
자기소개

숭실대학교
Soongsil University

이름 : 강승수

출생 : 1996.03.12

학교 : 숭실대학교

학과 : 스마트시스템소프트웨어

관심분야 : 백엔드 웹 개발

Part 2,
프로젝트1

기숙사 웹 어플리케이션

- [시스템 구성]
  - OS : Window
  - Language : Html, CSS, JavaScript
  - Framework : Bootstrap,Node.Js
  - Platform : MySQL
- [프로젝트 배경 및 목표]
  - 기숙사생들을 위한 웹 어플리케이션
- [구현 기능]
  - 로그인 기능
  - 외박 신청과 커뮤니티 기능이 있는 CRUD 게시판
- [프로젝트 영상]
  - https://www.youtube.com/watch?v=T-AqJQlyGUA&feature=youtu.be

기숙사 웹 어플리케이션

[본인 역할]

- Node.js를 이용한 로그인기능 구현
- Node.js를 이용한 게시판기능 구현
- DB를 연동하여 사용자 데이터 수집 및 조작

[프로젝트 진행에 대한 어려움 및 극복 방안]

어려움:

- 팀원들의 학년과 실력이 각자 차이를 갖으며 발생한 다양한 의견차이

해결방법 :

- 학년이 낮은 친구들과는 비교적 쉬운 HTML,CSS,JAVASCRIPT 스터디 진행.
- 학년이 높고 실력이 좋은 팀원에게는 보다 어려운 DB 스터디 진행.
- 주2회 4시간씩 업무 관련 회의를 하면서 진행도가 얼마가 되는 지 체크 및 개발 공유.

결과:

- 최상위 권의 실습 점수와 팀워크 및 팀장으로써의 역할에 대한 중요도 및 책임감 습득.

Part 2,

기숙사 웹 어플리케이션

로그인 기능

Login

- Login은 메인 페이지와 로그인 전용 페이지에서 가능하다. session 값을 이용해서 설정해주었다.
- 로그인 창에 Form 기능을 넣어 Node.js 파일로 보내주고, Student table에서 MySQL 기능을 사용하여 ID, PW가 있는지 검사한다.
  만약 틀렸을 경우, 틀렸다는 알림 창이 뜨고 다시 로그인 초기화면으로 돌아간다.
- 로그인을 하지 않은 상태에서는 커뮤니티, 학생생활, 마이페이지 창에 접근할 수 없다.
  만약 로그아웃상태에서 커뮤니티 창에 접근한다면, 커뮤니티 창이 아닌 로그인 창을 띄운다.

기숙사 웹 어플리케이션

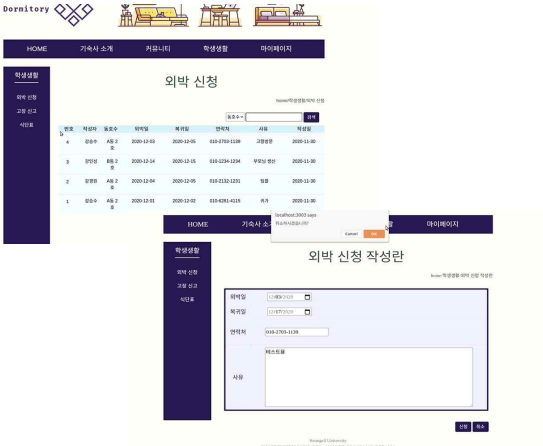
아이디 찾기 기능

ID/PW 찾기

- 자신의 이름과 동,호수를 입력하면 student table에서 My SQL을 사용하여 이에 맞는 학생의 ID와 비밀번호를 출력한다.
- 일치하는 회원 이름이 없을 경우,
  '일치하는 회원이 없습니다' 출력 후 로그인 초기화면으로 되돌아간다.
- 회원 이름은 일치하나 동, 호수가 맞지 않는 경우,
  '회원 정보가 주소가 일치하지 않습니다' 출력 후 로그인 초기화면으로 되돌아간다.
- 회원 이름과 주소가 모두 일치할 경우,
  회원 ID, PW를 알려주는 창이 뜬다.

기숙사 웹 어플리케이션

게시판 기능



게시판

- '글 작성' 버튼을 누르면 외박 신청 작성란 페이지가 보인다. Form 형식으로 node.js로 보낸 후 MySQL을 사용하여 DB에 저장한다.
- 글 작성 중 '취소'버튼을 누르면, '글을 취소하시겠습니까?'라는 알림 창이 뜨고 확인 버튼을 누르면 외박신청 목록 페이지로 돌아간다.
- 글 작성 후 '신청' 버튼을 누르면, '글을 등록하시겠습니까?'라는 알림 창이 뜬 후 외박신청 목록 페이지로 돌아간다. 목록에선 방금 작성한 글이 게시된 것을 볼 수 있다.

데이터베이스 연동

mysql > show tables;

Tables_in_web_programming
breakdown
sleepout
student

3 rows in set (0.00 sec)

〈Web Programming DB에 저장된 Table 목록〉

① Student : 회원정보 table

② Breakdown : 고장신고 table

③ Sleepout : 외박신청 table

mysql > select * from student;

id	password	email	name	dongho	birthday	home
20150297	1234	tmdtn0312@naver.com	강승수	A동 2호	1996년 3월 12일	전라남도 영광군 홍농읍 홍농로 546
20160348	kms0312	pfcskms1997@naver.com	강민성	B동 2호	1997년 3월 12일	강원도 춘천시 퇴계로 220-19
20180347	kang1234	kwkang98@naver.com	강경원	A동 2호	1998년 5월 30일	경기도 남양주시 도농동 부영아파트 201동
20180353	rlfekdud1139	rlfekdud1139@naver.com	길다영	B동 1호	1998년 3월 9일	경기도 안산시 단원구 고잔동 보네르빌리지 아파트

4 rows in set (0.00 sec)

mysql > select * from sleepout;

idx	name	dong	ho	homeout	homein	reason	calendar	telephone
1	강승수	A	2	2020-12-01	2020-12-02	귀가	2020-11-30	010-6281-4115
2	강경원	A	2	2020-12-04	2020-12-05	팀플	2020-11-30	010-2132-1231
3	강민성	B	2	2020-12-14	2020-12-15	부모님 생신	2020-11-30	010-1234-1234
4	강승수	A	2	2020-12-03	2020-12-05	고향방문	2020-11-30	010-2703-1139
5	길다영	B	1	2020-12-10	2020-12-18	테스트용	2020-11-30	010-2703-1139

5 rows in set (0.00 sec)

Part 2,
프로젝트 2

체온측정 및 얼굴인식을 통한 출입 차단 시스템

- [시스템 구성]
  - OS : Window, 라즈베리파이 OS
  - Language : Python,java,javascript
  - Framework : Bootstrap,Spring,OpenCV
  - Platform : MySQL

- [프로젝트 배경 및 목표]
  - '**체온 측정 및 얼굴인식을 통한 출입 통제**' 를 통한 전염병 발생시 해결방안 모색

- [구현 기능]
  - 체온 측정 기능
  - 얼굴인식 및 사원 판별 기능
  - 개폐 기능
  - 출입 사원 관리 기능
  - 마스크 착용 여부 판단 가능

Part 2,

체온측정 및 얼굴인식을 통한 출입 차단 시스템

[본인 역할]

- 라즈베리파이를 통한 사용자 센서데이터 취득 및 처리.
- 센서데이터를 통한 사원 출입문 개폐 통제 로직 개발.
- Spring Framework및 웹 어플리케이션을 통한 결과 출력.

[프로젝트 진행에 대한 어려움 및 극복 방안]

어려움:

- 카메라, 온도센서, 라즈베리파이 등 다양한 센서처리와 사용자 DB 데이터 연동에 대한 어려움.

해결방법 :

- 소켓 통신을 통한 센서데이터 공유방법 모색.
- 소켓 통신을 돕는 다양한 라이브러리 활용.

결과:

- 성공적인 센서데이터와 사용자 DB 데이터 연동 성공.
- 이 프로젝트를 통한 졸업작품 및 논문 연계.

Part 2,

체온측정 및 얼굴인식을 통한 출입 차단 시스템

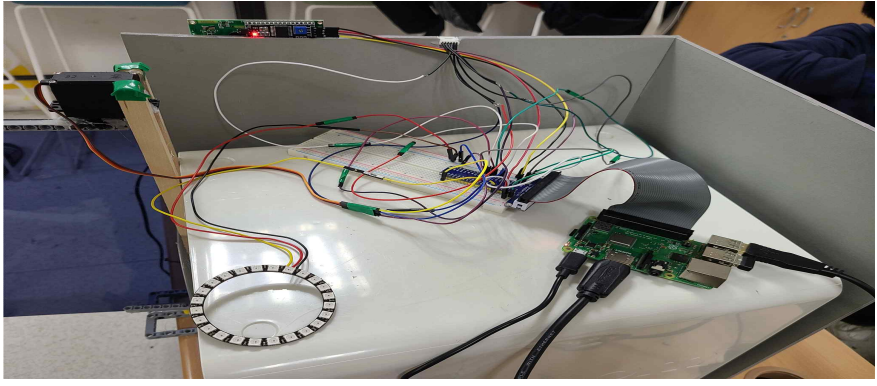
마스크 탐지 및 사원인식

- No Mask : 마스크를 착용하지 않음.
- Half Mask : 절반만 마스크 착용.
- Mask : 마스크를 잘 착용.
- 사용자(사원) 이름.
- DB에 없는 신원미상 출입자.
- : 머신러닝 오픈소스를 통한 Face detection

Part 2,

체온측정 및 얼굴인식을 통한 출입 차단 시스템

하드웨어

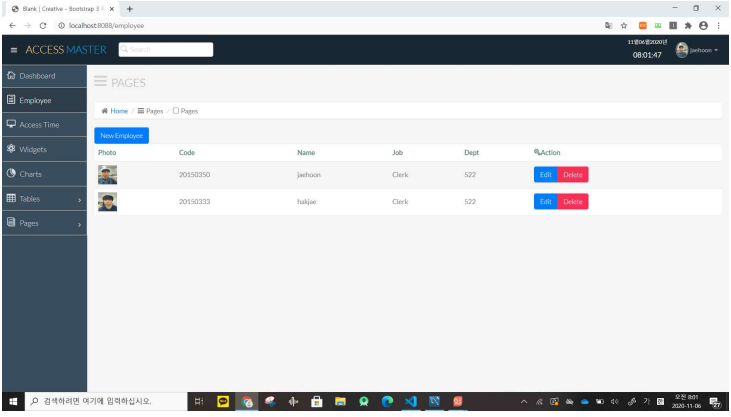


<라즈베리파이로 구현한 LED와 체온장치 및 개폐장치>

Part 2,

체온측정 및 얼굴인식을 통한 출입 차단 시스템

웹 애플리케이션



<스프링 프레임워크로 구현한 사원 출입 관리 목록>

Part 2,
프로젝트3

'친구의 역설' 증명

- [시스템 구성]
  - OS : Window
  - Language : python
- [프로젝트 배경 및 목표]
  - '자신의 친구'수 보다 '자신의 친구의 친구 수'가 더 많다는 '친구의 역설' 이론을 SNS 데이터를 분석해 증명
- [증명 내용]
  - Facebook Dataset에서 10만개의 데이터 추출 후 '자신의 친구 수'와 '자신의 친구들의 친구 수'의 평균을 비교
  - Twitter Dataset에서 10만개의 데이터 추출 후 Follow 관계에 따라 '친구'의 여러가지 개념을 정의 후 '자신의 친구 수'와 자신의 친구들의 친구 수' 평균을 구해서 비교
  - 두 경우 모두 '자신의 친구 수'보다 '자신의 친구들의 친구 수 평균'이 높다는 결과를 증명

Part 2, '친구의 역설' 증명

◦ [본인 역할]

- SNS dataset 획득 및 Sampling data 추출.
- SNS내 '친구' 개념 정의
- 파이썬을 통한 분석 코드 구현

◦ [프로젝트 진행에 대한 어려움 및 극복 방안]

어려움:

– Twitter에서 Follow의 형태에 따라 '친구'의 개념의 모호함

해결방법 :

– 서로 Follow한 경우, 한쪽만 Follow한 경우 등 부분적으로 세분화해서 결과를 도출

결과:

– '친구의 역설' 증명 및 최상위권의 프로젝트 실습 점수

Part 2, '친구의 역설' 증명

DATASET 및 Sampling Data

<Online Social Networks Research>

161	197	1214253495
161	198	1196463436
161	199	₩N
161	200	1216605917
161	201	1188281832
161	202	1213917386
161	203	1226329659
161	204	1224179610
161	205	₩N
161	206	1223994925
161	207	1228684802
208	209	₩N
208	210	₩N
208	211	1158460791
208	212	₩N
208	213	1208280272
208	214	1206899832
208	215	1161672209
208	216	1212348968
208	217	1207263030
208	218	1220840715

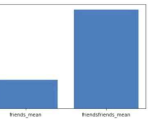
<Sampling Data>

'친구의 역설' 증명

Facebook데이터 분석

2) Facebook (친구 수, 친구의 친구 수의 평균 비교)

Friends Mean: 25.641822660871476 friendsfriends Mean: 88.0300634141058



Facebook의 경우, user1의 친구 목록에 user2가 있음에도, user2의 친구목록에는 user1이 없는 경우가 존재하였다. 이는 user2에 해당하는 사람이 조사대상이 아니기 때문에 발생하는 상황으로 보고, 평균을 구 하기 위해 user의 친구를 구하는 과정에서 user2에서 user를 찾아 user1을 친구로 보는 경우 도한 포함하 였다. 결과적으로 친구의 친구 수 평균이 친구 수의 평균보다 높아 Friendship Paradox 현상을 확인할 수 있었다.

<Facebook 데이터 분석1>

3) Facebook - Sampling

주어진 데이터를 사용하여 평균을 구하는 과정에서, sampling없이 진행하여 매우 많은 시간이 소요되었다. 따라서 Sampling을 통해 데이터를 분석하고, 비교해보기 위한 실험을 진행했다.

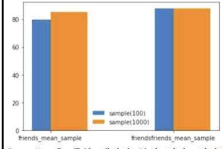
Ⅰ) user1에서 random하게 1만명 sampling

friends_sum : 7970 friends_mean : 79.7
friendsfriends_sum : 821726 friendsfriends_mean : 88.0300634141058

Ⅱ) user1에서 random하게 10만명

friends_sum2 : 85494 friends_mean2 : 85.494
friendsfriends_sum2 : 8823415 friendsfriends_mean2 : 88.0300634141058

sampling Ⅲ) 위의 두 경우 비교



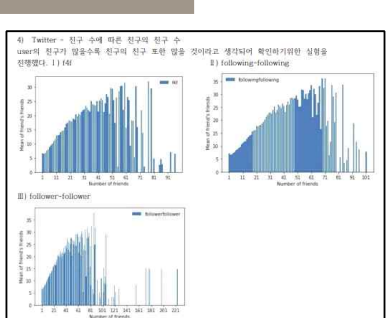
Sampling을 통한 데이터 분석 결과, 결과적으로 친구의 친구 수 평균이 친구 수의 평균보다 높은 Friendship Paradox 현상을 확인할 수 있었다. 하지만 Sampling을 거치지 않은 전체 데이터의 결과와는 차이를 확인할 수 있었다. 이는 전체 데이터를 통해 분석하는 과정에서 조사대상에 포함되지 않은 user2 또 한 user의 한 명으로서 계산하였지만, Sampling과정에서는 user1을 기준으로 sampling을 진행했기 때문에 user2를 user의 한 명으로서 계산할 수 없었고, 이에 따라 결과에 차이가 발생했다고 생각하였다. 이는 전체 데이터 분석 과정에서 user2를 한 명의 user로서 계산하지 않고 분석한다면 차이가 작아질 것이라고 예측하였지만 전체 데이터 분석 시 실행 시간 관계상 진행하지 못하였다.

<Facebook 데이터 분석2>

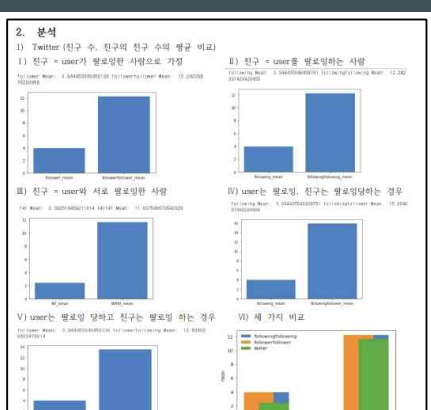
Part 2,

'친구의 역설' 증명

Twitter 데이터 분석



<twitter 데이터 분석1>



<twitter 데이터 분석2>

Part 3,
논문

무효 트랜잭션 필터링을 통한 블록체인 공간 효율성 향상 기법

- **[사용 시스템]**
  - OS : 리눅스
  - Language : Go
  - Framework : Docker
  - Platform : Hyperledger Fabric

- **[논문 배경 및 목표]**
  - **Order에서 비유효한 트랜잭션을 필터링해서 공간의 효율성을 늘린다.**

Part 3,

무효 트랜잭션 필터링을 통한 블록체인 공간 효율성 향상 기법

◦ [하이퍼레저 패브릭 동작원리]

① 여러 node에서 트랜잭션을 발생시키고 orderer에게 트랜잭션을 보낸다.
② orderer는 여러 트랜잭션을 받은 후 이를 정렬해서 블록을 생성한다.
③ 그 후 블록을 노드에게 다시 보낸다.

◦ [문제점]

- order에서 블록을 생성하는 과정에서 트랜잭션이 유효한지 비유효한지 판단하지 않는다.
- 유효한 트랜잭션은 블록안에 저장되는 것이 맞지만, 비유효한 트랜잭션까지 같이 저장되면서 공간을 차지하는 양이 늘어난다.

◦ [해결방법]

- orderer에서 미리 비유효한 트랜잭션을 필터링해서 제거해주면 공간의 효율성을 늘릴 수 있다.

무효 트랜잭션 필터링을 통한 블록체인 공간 효율성 향상 기법

무효 트랜잭션 필터링을 통한 블록체인 공간 효율성 향상 기법

이봉원°, 최재훈, 강승수, 김이주, 이은지*

숭실대학교 스마트 시스템 소프트웨어학과

ghddnjs1@naver.com, zpqls321@naver.com, tmdtn0312@naver.com, kimleju1408@gmail.com, eilee@ssu.ac.kr

요 약

블록체인 시스템은 다수의 노드가 원장을 공유함으로써 데이터의 높은 신뢰성을 제공하는 분산 데이터 저장 플랫폼이다. 본 논문에서는 최근 가장 각광을 받고 있는 블록체인 플랫폼 솔루션을 타겟으로 데이터 저장 시 불필요한 쓰기 증폭현상이 발생하는 점을 관찰하고 이를 해결하기 위한 기법을 제안한다. 구체적으로는 분산 환경에서 동시다발적으로 트랜잭션이 발생할 때 합의 과정에서 최종적으로 무효화되는 트랜잭션들이 블록체인에 유입됨으로써 공간낭비 및 성능저하를 발생시키는 점에 착안하여 무효 트랜잭션 필터링을 통한 블록체인 플랫폼 최적화 기술에 대해 연구하였다.

1. 서 론

최근 블록체인 기술은 데이터에 대한 높은 보안을 제공하는 분산 데이터 저장 시스템으로 크게 각광을 받고 있다. 블록체인 시스템은 다수의 노드가 거래내역 원장을 공유하고 각 노드는 머클트리(Merkle Tree)를 활용해 데이터를 체인 형태로 저장하여 데이터의 변경이나 조작을 사실상 차단시키는 데이터 저장 시스템이다. 이러한 특성은 과거 신뢰를 획득한 제3의 기관의 중재가 있어야만 가능했던 금융거래나 부동산 거래와 같은 분야에서 특정 기관 없이 참여자들에 의해서만도 신빙성 있는 거래를 성립시킬 수 있다는 장점이 있다. 비트코인 거래소는 이러한 블록체인의 기능을 실체적으로 입증한 대표적 사례로서 블록체인에 대한 전 세계적 관심을 불러 일으켰다.

블록체인은 플랫폼의 구성 노드로 참여가능한 대상의 허용범위에 따라 크게 퍼블릭 블록체인(Public Blockchain) 과 프라이빗 블록체인(Private Blockchain)으로 나뉜다. 퍼블릭 블록체인은 누구라도 블록체인 플랫폼을 구성하는 노드로 참여가 가능하며, 불특정 다수의 참여를 통해 일부 기관이나 개인이 플랫폼의 신뢰성을 해치지 못한다는 측면에서 이상적이다. 그러나 퍼블릭 블록체인은 구동환경이 불안정하고 운용능력을 확보하기 어려워 실제 환경에서 적용하기에는 한계가 존재한다. 이에 특정 기업이나 기관에서 허용된 대상만이 참여하는 프라이빗 블록체인 또는 하이브리드 블록체인이 실제 활용될 가능성이 높을 것으로 예상되고 있다.

하이퍼레저 패브릭은 대표적인 프라이빗 블록체인 플랫폼 솔루션으로 IBM의 적극적인 개발활동을 통해 현재 가장 기술적으로 성숙한 블록체인 플랫폼으로 인식되고 있다[1]. 하이퍼레저 패브릭은 블록체인 플랫폼이 성능과 정확성을 모두 갖출 수 있도록 다양한 형태의 최적화 기법을 적용해왔다. 예를 들어, 기존 퍼블릭 블록체인 시스템에서 합의 알고리즘으로 많이 사용해왔던 작업증명 (Proof-of-Work) 방식은 불필요한 계산연산을 통해 자원을 낭비한다는 지적을 받아왔다 또한 블록체인에서 포크 발생 시 데이터를 선택하는 알고리즘으로 Longest chain rule 이 많이 사용되어 왔는데 이 방식은 대규모의 동시실행성이 높은 시스템에서는 사실상 완벽한 합의를 이루기 어렵다는 지적을 받았다. 하이퍼레저는 이러한 기존 블록체인 시스템의 한계점을 극복하기 위해 특정 머신만이 블록체인에 참여하도록 허용하고 오더러(Orderer)라는 기존 분산 합의 시스템을 활용하여 다수의 노드 간에 정확한 합의를 도출해내도록 구성되었다.

그러나 본 논문에서는 현재 하이퍼레저의 구조가 동시다발적 트랜잭션 실행 과정에서 실패한 트랜잭션도 블록체인 시스템에 영구적으로 보관함에 따라 네트워크 내부의 쓰기 트래픽 및 스토리지 사용용량을 증가시킨다는 점을 관찰하였다. 이에 본 논문은 유효하지 않은 트랜잭션 데이터가 블록체인 저장 시스템에 삽입되는 것을 방지하여 불필요한 공간낭비 및 쓰기 트래픽 증가를 막는 "무효 트랜잭션 필터링 기법(Invalid Transaction Filtering)"을 제안한다.

본 논문의 구성은 다음과 같다. 2장에서는 기존 하이퍼레저패브릭의 데이터 저장 방식을 살펴보고 3장에서는 제안하는 무효 트랜잭션 필터링 기법에 대해 살펴본다. 4장에서는 결론을 맺고 향후 연구 방향에 대해 살펴본다.

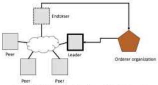


그림 1. 하이퍼레저 패브릭의 블록체인 플랫폼 구성

2. 하이퍼레저 패브릭의 데이터 저장구조

그림 1 은 하이퍼레저 패브릭의 구조를 단순화하여 보여준다. 하이퍼레저에서 블록체인 플랫폼을 구성하는 노드들은 모두 Peer 노드라고 불리는데 그 중 일부가 특수 역할을 수행하도록 선출된다. Endorser 노드는 사용자로부터 트랜잭션이 도착했을 때 트랜잭션이 유효한지를 시뮬레이션을 통해 판단하는 역할을 한다. 이것은 사전에 유효하지 않은 데이터가 블록체인 네트워크로 유입되는 것을 차단하는 역할을 수행하게 된다. Endorser 노드에 의해 검증을 받은 트랜잭션은 Endorser 에 의해 Orderer organization 으로 전달된다. Orderer organization 은 분산 시스템에서 동시다발적으로 발생하는 트랜잭션의 순서를 결정해줌으로써 동일한 버전으로 분산된 노드들이 합의를 이룰 수 있도록 해주는 역할을 수행한다. 하이퍼레저 패브릭은 Orderer 솔루션으로 Kafka[2] 나 Raft[2] 와 같은 분산 합의 시스템을 활용하고 있다. Orderer 는 트랜잭션들의 실행 순서를 결정하고 난 후 트랜잭션들은 해당 순서대로 적어둔다. 트랜잭션의 내용이 정해진 블록의 크기(예: 512KB) 만큼 채워지거나 사용자가 설정한 시간마다 주기적으로(예: 2 초) 블록을 생성하여 블록체인 노드들에게 전달한다. 이때 Orderer 로부터 블록을 받는 노드를 Leader 노드라고 하는데 Leader 노드는 동료 노드들에게 가십 프로토콜을 사용해 블록을 전달한다.

이 과정에서 하이퍼레저 패브릭은 아래와 같은 공간 낭비를 하게 된다. 첫째, Orderer 에서 트랜잭션의 순서를 지정함에 따라 충돌이 발생하는 트랜잭션들 중에는 반영에 실패하는 트랜잭션이 발생하게 된다. 이러한 트랜잭션들은 실제 블록체인 플랫폼의 데이터 상태와 무관함에도 불구하고 영구적으로 블록체인에 저장되어 공간 낭비를 초래하게 된다. 둘째, 하이퍼레저 패브릭은 블록이 모두 채워지지 않았더라도 체인을 동일한 크기의 블록으로 관리하기 때문에 빈 영역을 NULL 값으로 채워 블록을 전달한다. 트랜잭션 커밋을 자주 발생시켜야 하는 시스템의 경우 이 주기가 매우 짧아 심각한 공간 낭비 상황이 발생할 수 있다.

3. 무효 트랜잭션 필터링을 통한 공간효율성 향상

본 논문에서는 앞서 설명한 문제점을 해결하기 위해 블록이 Peer 노드에 전파되기 전에 무효한 트랜잭션을 검사하고 해당 데이터를 블록에서 삭제하는 무효 트랜잭션 필터링 기법을 제안한다. 현재 하이퍼레저 패브릭에서는 Orderer 가 트랜잭션 간의 순서를 결정하면 Leader 노드가 각 트랜잭션을 해당 순서로 실행하면서 유효성 검증을 하게 된다. 이 때 무효한 트랜잭션으로 판명나는 경우 블록에 포함시키지 않도록 하여 공간 사용량 및 네트워크 트래픽을 감소시키고자 한다. 이러한 기법은 센서 데이터 반영과 같은 트랜잭션의 동시실행성이 높은 환경에서 성능을 향상시키는 데에 크게 기여할 것으로 기대된다. 본 연구의 향후 연구로 실제 워크로드를 하이퍼레저 패브릭에 실행할 때 필터링 기법을 통한 공간 절감효과를 조사하고, 가변 크기의 블록으로 체인을 구성할 때의 문제점 등을 분석하여 실용적이고 효율적인 블록체인 플랫폼 기술을 완성하고자 한다.

3. 결론

본 논문에서는 대표적인 블록체인 플랫폼인 하이퍼레저 패브릭에서 데이터를 저장하는 플로우를 살펴보고 그 과정에서 발생하는 공간 비효율성 문제를 분석하였다. 향후 이를 개선할 수 있는 무효 트랜잭션 필터링 기법을 구현하여 실제 블록체인 시스템의 성능 개선을 이루고자 한다. 또한 불필요한 데이터를 디바이스 레벨에서 삭제하여 상위 응용에서는 단순한 형태로 데이터를 저장하더라도 공간낭비를 최소화할 수 있는 디바이스 인터페이스 및 저장기술에 대해 연구하고자 한다.

본 연구는 과학기술정보통신부 및 정보통신기획평가원의 SW 중심대학사업과 교육부의 이공분야 기초연구사업의 연구결과로 수행되었음(2018-0-00209, NRF-2019R1A2C1090337)

참고문헌

[1] https://www.hyperledger.org/projects/fabric
[2] https://kafka.apache.org/
[3] https://raft.github.io/

Part 4,
SSAFY

삼성 청년 SW 아카데미

THANK YOU