일반논문 (Regular Paper)
방송공학회논문지 제26권 제3호, 2021년 5월 (JBE Vol. 26, No. 3, May 2021)
https://doi.org/10.5909/JBE.2021.26.3.321
ISSN 2287-9137 (Online) ISSN 1226-7953 (Print)

빠른 검색을 위한 음원 시그니처 인덱싱 방법

김 상 균[a)‡], 이 경 식[b)]

Music Source Signature Indexing Method for Quick Search

Sang-Kyun Kim[a)‡] and Kyoung-Sik Lee[b)]

요 약

블록체인은 자본 거래나 보안 데이터의 안전한 전송을 위한 플랫폼으로 그 가치가 높아지고 있다. 아울러 블록체인은 동영상, 음악, 사진과 같은 대용량의 데이터를 안전하게 저장하고, 거래 내용이나 서비스 이용 명세 등을 안전하게 관리할 수 있는 새로운 플랫폼으로서의 가능성을 가지고 있다. 블록 내 대용량 미디어 데이터를 저장할 수 없기에 분산저장 시스템(IPFS)과 음원 시그니처 데이터의 해시 정보를 이용하여 블록 내 음원 정보를 저장하고, 저장된 음원 데이터를 검색하는 속도에 관한 연구가 진행되었다. 본 논문에서는 기존 연구가 제시했던 검색 속도를 향상시킬 수 있는 블룸필터를 이용한 음원 시그니처 인덱싱 방법을 제안한다. 실험 결과 기존 검색 성능(O(n))보다 향상된 검색 성능 (O(1))을 달성할 수 있음을 확인할 수 있었다.

Abstract

Blockchain is increasing in value as a platform for safe transmission of capital transactions or secure data. In addition, blockchain has the potential as a new platform that can safely store large amounts of data such as videos, music, and photos, and safely manage transaction details and service usage specifications. Since it is not possible to store large-capacity media data in a block, research on the performance of storing sound source information in a block and retrieving the stored sound source data by using the distributed storage system (IPFS) and the hash information of the sound source signature data was conducted. In this paper, we propose a sound source signature indexing method using a bloom filter that can improve the search speed suggested by previous studies. As a result of the experiment, it was confirmed that improved search performance (O(1)) than the existing search performance (O(n)) can be achieved.

Keyword : blockchain, bloom filter, music search and retrieval, IPFS, audio signature

a) 명지대학교 융합소프트웨어학부(Department of Convergence Software Myongji University)
b) 명지대학교 컴퓨터공학과(Computer Engineering Department Myongji University)
‡ Corresponding Author : 김상균(Sang-Kyun Kim)
E-mail: goldmunt@gmail.com
Tel: +82-31-330-6443
ORCID: https://orcid.org/0000-0002-2359-8709

※ This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. 2019R1F1A1041882).
· Manuscript received April 13, 2021; Revised May 20, 2021; Accepted May 20, 2021.

Ⅰ. 서 론

최근 개인의 PC환경 혹은 모바일 기기를 통해 손쉬운 미디어 콘텐츠의 제작과 소비를 할 수 있게 되었다. 이에 따라 기존의 TV, 라디오와 같은 전문적인 대중 매체보다 개인 미디어의 상업성 및 중요성이 높아지고 있다. 특히, 개인 미디어의 대표적 플랫폼인 유튜브의 경우, 개인이 제작한 콘텐츠에 광고를 삽입하여 수익 창출이 가능한 서비스를 제공하고 있다[1]. 이러한 서비스를 통해 유튜브는 2017년 전체 국내 광고시장의 37%를 차지하였으며, 2018년에는 그 비중이 40%로 증가하였다[2].

다양한 플랫폼을 통해 엄청난 양의 미디어 콘텐츠가 생성·소비되면서 저작권과 관련된 문제가 대두되고 있다. 특히 음원의 경우, 동영상을 비롯한 거의 모든 미디어 콘텐츠 제작의 필수 요소로 활용되고 있으므로, 음원의 사용과 관련된 분쟁은 더욱 치열하다. 이를 해결하고자 인터넷 방송 플랫폼인 트위치에서는 콘텐츠 제작에 사용된 음원에 대해 실시간 감시를 진행하고 있다[3]. 유튜브에서는 음원 저작권을 위반하여 콘텐츠를 제작하는 경우, 이로 인해 발생한 수익을 콘텐츠 창작자가 배분받지 못하게 된다[4]. 따라서 음원의 권리자가 음원을 포함한 콘텐츠 배포 및 이용으로 인해 발생하는 수익을 할당받을 수 있는 근본적인 해결책이 요구되고 있다.

음원의 저작권을 효과적으로 관리하기 위하여, 음원 파일과 그 특징을 담은 시그니처 파일을 블록체인을 통해 저장하는 방법과 저장된 다량의 데이터에 대하여 효과적으로 검색하는 방법에 관해 연구하고, 그 결과를 분석하는 것이 필요하다.

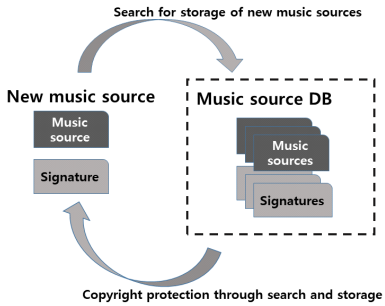

그림 1. 음원 및 시그니처 파일의 저장과 검색 간의 관계
Fig. 1. The relationship between saving and retrieving music and signature files

그림 1은 저작권 보호를 위한 음원 및 시그니처 파일의 저장과 검색 간의 관계를 보여주는 그림이다. 신규 음원을 음원 데이터베이스에 저장하기 위한 과정을 예로 들 수 있다. 신규 음원을 저장하기 위해 기존의 음원 데이터베이스를 검색하여 동일한 음원이 있는지의 결과를 알 수 있다. 이를 기반으로 신규 음원을 데이터베이스에 저장함으로써 해당 음원의 저작권을 보호할 수 있다.

이전 연구에서는 음원 시그니처를 20바이트로 축소하여 이를 블록체인과 분산저장 시스템(IPFS)에 저장하는 가능성에 대해 고찰하였고[5], 이를 근거로 블록체인과 분산저장 시스템을 활용한 음원 및 시그니처의 저장 방법을 제안하였으며[6], 저장된 다수의 음원 시그니처 데이터들에 대하여 특정 질의 시그니처의 검색 시간을 측정하였다[7]. 본 연구는 블록체인을 이용한 멀티미디어 서비스 플랫폼 내 효율적인 미디어 저장 및 관리, 빠른 미디어 검색, 미디어 사용자 및 저작자의 권리 추적 등의 연구에 기여할 것으로 예상된다. 본 논문에서는 검색 효율의 개선을 위해 블룸필터를 활용한 음원 시그니처 인덱싱 방법을 제안하고, 실험을 통해 검색 효율성을 평가 분석한다.

본 논문의 구성은 다음과 같다. 2장에서는 블룸필터의 원리를 이해한 후 블룸필터를 이용한 시그니처 인덱싱 방법을 설명한다. 3장에서는 블룸필터 인덱싱 방법을 이용하여 음원 시그니처를 검색할 때의 검색 시간 실험 결과를 분석한다. 마지막 4장에서는 결론을 도출하고 향후 연구 방안을 제시한다.

Ⅱ. 블룸필터를 활용한 검색 시간 단축 방안

1. 블룸필터의 원리

블룸필터는 필터링 과정을 거쳐 탐색 대상이 되는 데이터의 개수를 줄임으로써 빠른 데이터의 검색이 가능하도록 설계된 자료구조이다[8-10]. 블룸필터는 기본적으로 m 비트 크기의 배열로 구성되어 있으며, 저장하고자 하는 데이터에 대하여 k개의 서로 다른 해시 함수를 거친 후 각 해시 함수의 결과를 m비트 크기의 블룸필터에 비트맵 형식으로 저장한다.

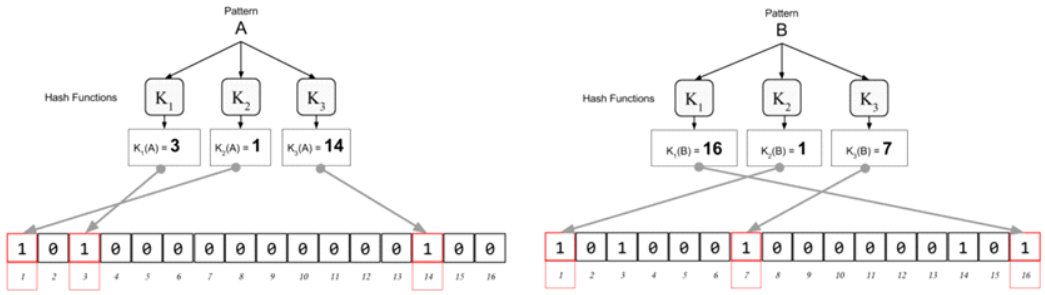

그림 2. 블룸필터를 통한 인덱스 추출 예
Fig. 2. Example of index extraction through bloom filter

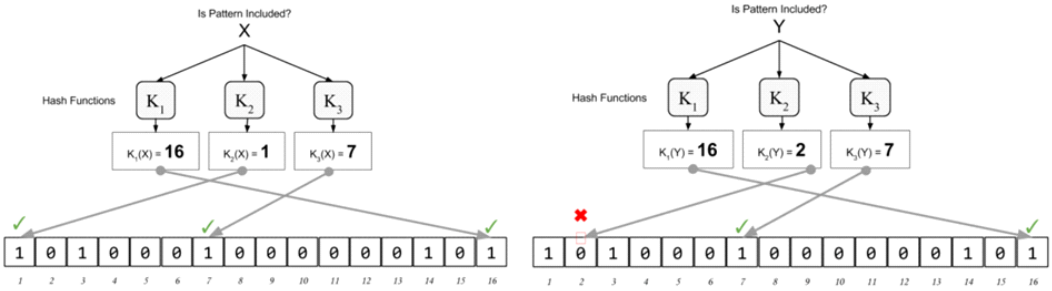

그림 3. 블룸필터를 통한 인덱스 필터링 예
Fig. 3. Example of index filtering through bloom filter

그림 2는 블룸필터를 통해 특정 데이터로부터 블룸필터를 통해 인덱스를 생성하는 예시를 설명한다[8][9]. 그림 2의 예시에서는 데이터 'A'에 대하여 K1, K2, K3 이상 3개의 해시 함수를 적용하였다. 이에 관한 결과로 각각 3, 1, 14의 해시값을 얻었고, 이는 16비트로 설정된 블룸필터의 3번째, 1번째, 14번째 배열에 1의 값을 가지며 블룸필터 인덱스로 저장된다. 다시 'B'라는 데이터의 인덱스를 추출하기 위해 같은 방식으로 K1(), K2(), K3() 이상 3개의 해시 과정을 거친 후 각각 16, 1, 7의 해시값을 얻었다. 이를 토대로 블룸필터의 16번째, 1번째, 7번째 배열의 값을 1로 변경하여 블룸필터 인덱스로 저장한다. 1번째 배열의 값이 이미 1의 값을 가지고 있더라도 블룸필터에서는 문제가 되지 않으며, 최종적으로 '1010001000000101'의 16bit 블룸필터 값을 얻을 수 있다.

그림 3은 그림 2의 과정을 통해 생성된 블룸필터 인덱스를 이용하여 새로 질의되는 특정 데이터의 존재 여부를 검색하는 과정을 설명한다[8][9]. 검색하고자 하는 데이터 'X'에 대해 K1, K2, K3 이상 3개의 해시 과정을 거쳐 각각 16, 1, 7의 해시값을 얻었다. 만약, 각 해시값이 가리키는 블룸필터 배열의 값이 모두 1이라면 'X'라는 데이터는 블룸필터를 통과하게 된다. 하지만 3개의 해시값이 가리키는 블룸필터 값 중 하나라도 0이라면 해당 데이터는 데이터베이스 내에 존재하지 않는다는 것을 의미한다.

블룸필터를 통과하는 과정을 거치게 되면 확실히 존재하지 않는 데이터에 대한 불필요한 탐색시간을 줄일 수 있으므로 검색 효율을 높일 수 있다. 그러나 단순 필터링 과정만으로는 음원 데이터베이스의 크기에 선형적으로 비례하여 증가하는 검색 시간을 해결하기에는 어려움이 있다.

2. 블룸필터 인덱싱을 활용한 시그니처 파일의 검색 방법

블룸필터는 '0'과 '1'의 조합으로 모든 필터의 값을 표현한다. 각 음원 시그니처 파일의 해시값은 블룸필터를 통과

한 이진 형태의 고유 인덱스로 변환할 수 있다. 그림 2의 예시에서 데이터 'A'와 'B'를 의미하는 블룸필터 고유 인덱스값은 각각 '1010000000000100', '1000001000000001'이 된다.

생성된 인덱스들을 이진 트리의 형태로 구성하고[11], 리프노드(leaf node)가 각 인덱스에 해당하는 음원 시그니처 파일의 IPFS 해시값을 가리키도록 한다[12]. 이 때 음원 시그니처 파일의 IPFS 해시값은 블록체인 내 블록에 저장되어 있다. 시그니처 파일의 검색 시에는 이진 트리의 탐색을 통해 원하는 시그니처 파일의 IPFS 해시값에 접근하여 분산 저장 시스템에 저장된 원본 시그니처 파일로의 접근이 가능하다[12]. 이로 인해, 음원 시그니처 파일의 개수가 증가하더라도 이진 트리로 구성된 인덱스 검색을 통해 신속한 음원(시그니처) 파일의 검색이 가능하다[11].

3. 블룸필터 해시 함수 결정

블룸필터 인덱싱을 활용한 시그니처 파일의 검색 시간을 측정하기 위해, 검색 대상이 되는 시그니처 파일의 개수를 1개부터 1,000개까지 순차적으로 증가시키면서 특정 질의 음원에 대한 검색 시간을 측정한다.

먼저 블룸필터를 구성하는 최적의 해시 함수를 결정하기 위하여 해시 연산의 결과를 얻기까지 소요된 시간과 블룸필터 인덱스를 생성하기까지 소요된 시간을 측정하였다. 실험은 6.6MB 크기의 시그니처 파일 한 개를 대상으로 한 번의 해시 연산만을 적용하였으며, 블룸필터 인덱스 생성 시에도 한 번의 해시 연산만을 적용하였다. 비교 대상이 되는 해시 함수로는 연산 속도가 가장 빠른 열 개의 해시 함수를 선정하였다.

그림 4에서 파란색 실선은 한 개의 시그니처 파일에 대하여 특정 해시 함수를 통해 해시 연산의 결괏값을 얻기까지 소요된 시간(초)이며, 주황색 실선은 동일한 해시 함수를 적용하여 블룸필터 인덱스를 생성하기까지 소요된 시간(초)을 나타낸다.

측정 결과, 블룸필터 인덱스의 생성 속도는 선택한 해시 함수의 연산 속도에 비례한다는 결과를 보여주었다. 이에 블룸필터 인덱스 생성을 위한 최적의 해시 함수 선택의 조건으로 연산 속도가 빠른 해시 함수의 선택이 유리할 것으로 판단하여 해시 연산 속도가 가장 빠른 crc32b, crc32, md4 이상 세 개의 해시 함수를 선택하였다.

블룸필터를 통해 생성되는 이진 인덱스의 크기는 초기 설정한 블룸필터의 크기에 따라 결정되며, 이진 인덱스 생성에 사용한 해시 함수의 개수만큼 '1' 값이 포함된다. 때문에, 블룸필터를 통해 표현 가능한 인덱스의 개수는 블룸필터 배열의 크기 m과 사용한 해시의 개수 k와의 중복 조합(Homogeneous combination) 연산인 ${}_mH_k$로 계산할 수 있다[13].

시그니처 파일의 검색 시간 측정 실험에서 대상이 되는 최대 시그니처 파일의 개수가 1,000개이므로, 블룸필터의 배열 크기는 ${}_mH_3 \geq 1{,}000$을 만족하는 최소 크기인 20bit부터 10bit씩 증가시키며 검색 시간을 측정하였다.

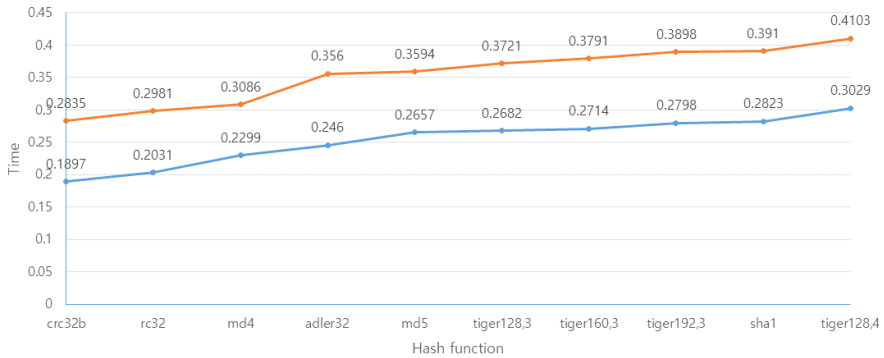

그림 4. 해시 연산 속도와 블룸필터 인덱스 생성 속도 비교
Fig. 4. Comparison of hash operation speed and bloom filter index creation speed

Ⅲ. 블룸필터를 이용한 음원 시그니처 인덱싱 및 검색 시간 비교 실험

표 1은 각각 블룸필터의 크기를 각각 20bit부터 50bit까지 10bit씩 늘려가며 검색 시간을 측정한 결과이다.

표 1. 블룸필터 크기 변화에 따른 검색 시간 비교
Table 1. Comparison of search time according to change in bloom filter size

# of queries	Search time (s) (20bit filter)	Search time (s) (30bit filter)	Search time (s) (40bit filter))	Search time (s) (50bit filter)
1	1.21	1.22	1.21	1.21
100	1.23	1.21	1.23	1.25
200	1.20	1.23	1.26	1.24
300	1.26	1.21	1.19	1.28
400	1.24	1.20	1.23	1.26
500	1.24	1.22	1.22	1.24
600	1.66	1.20	1.21	1.23
700	1.22	1.22	1.24	1.24
800	2.05	1.75	1.22	1.23
900	1.68	1.68	1.23	1.23
1000	1.59	1.63	1.26	1.26

블룸필터의 크기가 20bit인 경우, 저장된 시그니처 파일의 개수가 각각 600개, 800개, 900개, 1,000개일 때 평균 1.2초~1.3초를 초과하는 검색 시간이 소요되었다. 블룸필터의 크기가 30bit인 경우, 저장된 파일의 개수가 800개, 900개, 1,000개일 때 평균 1.2초~1.3초의 범위를 초과하였다. 블룸필터의 크기를 40bit와 50bit로 설정하였을 때 모든 구간에서 1.2초~1.3초의 균등한 검색 시간을 보였다. 이는 검색 대상 파일의 개수에 비례하여 선형적으로 증가하던 기존의 검색 방법[7]과 비교해 블룸필터를 사용한 방법이 효과적인 검색 방안이 될 수 있음을 의미한다. 이전 검색 방법이 데이터베이스 크기(n)가 증가하면 검색 시간도 비례해서 증가하는 복잡도 O(n)이었던 것에 비해, 블룸필터 인덱싱 방법을 이용하였을 때 복잡도가 O(1)으로 감소하는 데 기인한다고 볼 수 있다. 주목할 것은 제안하는 블룸필터 인덱싱 방법이 데이터베이스 크기(n)가 아닌 블룸필터 배열의 크기(m)에 의해 검색 속도가 변화하는 것이다. 다시 말해 블룸필터 배열의 크기(m)가 20, 30, 40, 50bit로 증가해도 이는 이진 트리 검색 복잡도인 O(log m) ≈ O(1)와 같다는 것이다.

블룸필터의 크기가 20bit, 30bit일 때 검색 시간이 늘어나는 원인은 복수 개의 시그니처 파일이 동일한 블룸필터 인덱스를 할당받아, 추가로 해당 시그니처 파일들을 직접 비교하는 과정에서 탐색시간이 증가하는 것으로 파악된다. 중복되는 블룸필터 인덱스 생성을 방지하기 위해서는 저장 데이터의 개수에 비하여 블룸필터의 크기를 충분히 크게 생성할 필요성이 있다. 참고로 블룸필터의 크기가 20bit인 경우 생성 가능한 최대 인덱스의 개수는 1,540($_{20}H_3$= $_{22}C_3$)개이며, 블룸필터의 크기가 30bit, 40bit, 50bit인 경우에는 각각 4,960($_{30}H_3$=$_{32}C_3$)개, 11,480($_{40}H_3$=$_{42}C_3$)개, 22,100($_{50}H_3$=$_{52}C_3$)개의 인덱스 생성이 가능하다[13]. 실험 결과 1,000개의 시그니처 파일의 중복 인덱스 생성을 피하기 위해서는 필터 크기가 30bit(인덱스 개수: 4,960) 보다는 크며 40bit(인덱스 개수: 11,480)와 유사해야 한다.

Ⅳ. 결론 및 향후 연구 과제

음원 시그니처 파일의 선형적 검색 속도 개선을 위해 블룸필터 인덱싱을 활용한 검색 방법을 제안하였다. 검색 실험 결과 기존 연구 결과[7]의 검색 시간 복잡도(O(n))에 비교해 향상된 검색 효율(O(1))을 확인할 수 있었다. 이를 통해 블룸필터 인덱싱은 음원 검색과 같은 다수의 고용량 데이터에 대하여 효율적인 검색 방법이 될 수 있음을 검증하였다.

본 논문에서는 음원이나 시그니처 파일의 변조 가능성에 대해서는 배제한 후 연구를 진행하였다. 이에, 음원이나 시그니처 파일의 변조에 대응할 수 있는 시그니처 추출 방법이나 비교 방법에 관한 추가적인 연구가 진행될 예정이다. 아울러 블록체인과 분산저장 시스템을 활용한 음원 저장 및 검색 서비스가 기존 인터넷이나 앱을 통한 음원 저장 및 검색 서비스에 비해 어떠한 차별점을 갖는지에 관한 연구가 필요하다.

참 고 문 헌 (References)

[1] Youtube advertising service, https://www.youtube.com/intl/ko/ads/?&subid=kr-ko-ha-yt-bk-c-plt!o3~CjwKCAjw0N3nBRBvEiw

AHMwvNrnMfFqjLUOGS-OMsm8jSxcjPkAIx9Mon7z3BRJc_Ad98wAPoh5iTBoCBokQAvD_BwE~%7badgroup%7d~kwd-463486203081~1727016377~337034874168&gclid=CjwKCAjw0N3nBRBvEiwAHMwvNrnMfFqjLUOGS-OMsm8jSxcjPkAIx9Mon7z3BRJc_Ad98wAPoh5iTBoCBokQAvD_BwE&gclsrc=aw.ds (accessed May. 1, 2019).

[2] 2018 1st half sector analysis report, https://www.slideshare.net/MezzoMedia/2018-106564325 (accessed May. 1, 2019).

[3] Twitch music copyright guide, https://www.twitch.tv/p/ko-kr/legal/community-guidelines/music/ (accessed May. 1, 2019).

[4] Youtube music copyright guide, https://creatoracademy.youtube.com/page/lesson/artist-copyright?hl=ko (accessed May. 1, 2019).

[5] Kyoung-Sik Lee, Sang-Kyun Kim, "Use of blockchain for music content copyright protection," *KIBME Conference in summer*, pp.295-299, 2019.

[6] Kyoung-Sik Lee, Sang-Kyun Kim, "Music source and signature storage method using blockchain and distributed storage system," *Journal of Broadcast Engineering*, Vol. 24, No. 6, Nov. 2019.

[7] Kyoung-Sik Lee, Sang-Kyun Kim, "Analysis of Storage and Retrieval Results of Audio Sources and Signatures using Blockchain and Distributed Storage System," *Journal of Broadcast Engineering*, Vol. 24, No. 7, Dec. 2019.

[8] Antonopoulos, Andreas M., *Mastering Ethereum*, Oreilly & Associates Inc., 2018.

[9] Antonopoulos, Andreas M., *Mastering Bitcoin*, Oreilly & Associates Inc., 2014.

[10] Biplob Debnath, Sudipta Sengupta, Jin Li, David J. Lilja, David H.C. Du, "BloomFlash: Bloom Filter on Flash-Based Storage," *IEEE 31st International Conference on Distributed Computing Systems*, 2011.

[11] Tobin J. Lehman, Michael J. Carey, "A Study of Index Structures for Main Memory Database Management Systems," *Computer Sciences Technical Report*, pp.605, 1985.

[12] IPFS Documentation, https://docs.ipfs.io/ (accessed May. 1, 2019).

[13] Yeonwha Choi, *A Instruction Method of Duplicate Combination Using Occupation Problem*, Graduate School of Seoul National University, 2011.

저 자 소 개

이 경 식

- 2011년 ~ 2017년 : 명지대학교 컴퓨터공학과 학사
- 2017년 ~ 2020년 : 명지대학교 일반대학원 컴퓨터공학과 석사
- ORCID : https://orcid.org/0000-0003-1308-2650
- 주관심분야 : 4D media, audio contents, VR and Internet of Things, Blockchain

김 상 균

- 1997년 : 아이오와대(Univ. of Iowa) 전산과학, B.S.(1991), M.S.(1995), Ph.D(1997)
- 1997년 3월 ~ 2007년 2월 : 삼성종합기술원 멀티미디어랩 전문연구원
- 2007년 3월 ~ 2016년 2월 : 명지대학교 컴퓨터공학과 교수
- 2017년 3월 ~ 현재 : 명지대학교 융합소프트웨어학부 데이터테크놀로지전공 교수
- ORCID : https://orcid.org/0000-0002-2359-8709
- 주관심분야 : digital content(image, video and music) analysis and management, 4D media, blockchain, VR, Internet of Things and multimedia standardization, metaverse