LSH 기반 유사성 해시 방법 분석

김윤정, 이만희*
한남대학교, *한남대학교
kyunjeong125@gmail.com, *manheelee@hnu.kr

A Survey on Similarity Hashing Scheme based LSH

Yun-Jeong Kim, Man-Hee Lee*
Hannam Univ., *Hannam Univ.

요 약

파일 유사도 측정 방법은 악성파일 분류, 파일 변조 및 중복 탐지 등 많은 분야에 사용되고 있다. 파일 유사도 측정 방법 중 해시 기반 유사도 평가는 간단하고 빠른 측정 방법이며, 많은 데이터 간 유사성을 비교하는데 유용하다. LSH(Locality Sensitive Hashing) 기반 유사성 해시는 해시 충돌을 극대화하여 유사한 데이터일수록 유사한 해시값을 갖는 해싱 기법이다. 이 해싱 기법은 입력값이 한 글자라도 다르면 완전히 다른 해시값을 출력하는 암호학적 해시 함수와 달리, 극대화된 해시 충돌을 이용하여 유사한 해시값을 찾아 비교군과의 유사한 정도를 수치로 나타낼 수 있어 파일의 유사도 측정에 적합하다. 본 논문에서는 LSH와 LSH 기반 유사성 해시 3가지(ssdeep, nilsimsa, TLSH)의 성능을 비교·분석한다. 분석 결과 해시값과 유사도 점수 생성 시간에서 nilsimsa가 가장 많은 시간을 소요하였고, 유사도 평가의 정확도 측면에선 ssdeep이 다른 LSH 기반 해시보다 세세한 차이를 판단하기 어려운 성능적 한계가 존재했다. 즉, 해시값 및 유사도 점수 생성 시간과 유사도 평가의 정확도에 있어 TLSH가 가장 뛰어남을 확인하였다.

Ⅰ. 서론

파일 유사도 측정 방법은 악성파일 분류, 파일 변조 및 중복 탐지 등 많은 분야에 사용되고 있다. 파일 유사도 측정 방법 중 해시 기반 유사도 평가는 바이너리 및 텍스트 비교에 비해 간단하고 비교 속도가 빠른 측정 방법으로, 많은 데이터 간 유사성을 비교하는데 유용하게 사용된다[1].

SHA-1, MD5와 같은 암호학적 해시 함수는 입력값이 한 글자라도 다르면 완전히 다른 해시값을 출력하므로 데이터 유사성을 판단하는 데 한계가 있다. 이를 해결하기 위해 근사 일치 방법을 이용한 퍼지 해시라고도 불리는 AHBM(Approximate Hash Based Matching) 알고리즘이 제안되었다[2]. 퍼지 해시에 속하는 LSH 기반 유사성 해시는 해시 충돌을 극대화하여 유사한 데이터일수록 유사한 해시값을 갖는다. 또한, 파일의 유사한 정도를 수치로 나타낼 수 있어 파일의 유사도 측정에 적합하다.

본 논문에서는 파일 유사도 비교에 유용한 LSH 기반 유사성 해시의 특징을 소개하고 알고리즘 성능을 비교·분석한다. 2장은 4가지의 대표적인 LSH 기반 유사성 해시의 알고리즘을 간략히 소개하고, 3장은 파일의 유사도 측정 실험을 수행하여 각 알고리즘의 성능을 측정하고 결과를 분석한다. 끝으로 4장에서는 본 논문의 결론을 맺는다.

Ⅱ. LSH 기반 유사성 해시

2.1 LSH(Locality Sensitive Hashing)

LSH는 비슷한 항목을 같은 버킷에 넣는 해시 알고리즘으로, 이를 통해 다양한 유사성의 정도를 식별할 수 있다. 또한, 고차원 데이터를 상대적 거리를 유지하며 저차원으로 축소할 수 있다[3]. LSH는 문서, 오디오, 비디오 등 모든 종류의 데이터에 적용할 수 있어 다음과 같은 응용 프로그램에 활용할 수 있다[4].

- 문서, 웹페이지 등의 컨텐츠 중복 탐지
- 게놈(genome) 데이터베이스에서의 유사 유전자 발현 식별
- 대규모 이미지 검색 및 순위 책정
- 오디오 및 비디오 핑거프린팅(fingerprinting)

2.2 ssdeep

ssdeep은 대표적인 퍼지 해시로서, Kornblum가 2006년 CTPH (Context Triggered Piecewise Hash)와 함께 제안하였다. 기존의 퍼지 해시는 데이터를 고정된 크기로 나누어 계산하는 방식인데, 입력값에 변화가 생기면 분할된 내용이 크게 달라져 해시값에 큰 영향을 주는 문제가 있다. CTPH 방식은 이러한 문제 해결을 위해 데이터의 문맥(context)을 구분할 수 있는 경계를 정하고, 문맥을 기준으로 나누어 해시를 계산한다.

계산 결과인 ssdeep 해시값의 길이는 최대 148바이트이다. 또한, 유사도 점수는 0부터 100까지의 정수값으로 표현되며, 유사할수록 100에 가깝고 유사하지 않을수록 0에 가깝다[5]. ssdeep은 악성코드 간 유사도를 빠르게 식별하는 것이 가능하여 악성코드 유사도를 비교하고 분류하는 분야에 주로 활용되었다[6].

2.3 nilsimsa

nilsimsa는 문자 또는 문자열의 트리그램(Trigram)을 생성하는 슬라이딩 윈도우(sliding window)를 사용하며, 32바이트 출력값을 가지는 LSH 기반 알고리즘이다. 이 알고리즘은 효과적인 스팸 메일 탐지를 목적으로 고안되었으며, 실제로 nilsimsa을 이용하여 스팸 메시지의 유사성을 식별하여 스팸 메일을 퇴치하기 위한 연구가 진행된 바 있다[7].

nilsimsa 유사도는 해시값의 비트를 비교하여 같은 위치에 같은 비트가 많을수록 큰 값을 갖는다. 유사도 점수는 0부터 128까지의 정수값으로 표현되며, 128에 가까울수록 유사하다고 판단한다. 이를 기반으로 악성코드 유사도를 계산하고 악성코드 유형을 예측하는 연구에 활용되었다[8].

2.4 TLSH(Trend micro Locality Sensitive Hashing)

TLSH는 2013년 Trend Micro 사가 LSH를 개량하여 만든 해시 알고리즘이다. TLSH는 크기 5의 슬라이딩 윈도우로 데이터를 분할하고, 분할된 데이터를 4등분하여 설정한 경계를 기준으로 헤더와 바디를 계산한다. 그 후 헤더와 바디를 이어 70바이트의 해시값을 구성한다. 두 TLSH 해시값의 유사도 정도를 정수값으로 나타낼 수 있고, 이를 TLSH 유사도 점수라

$$mod\_diff(X,Y,R) = MIN((X-Y),(Y-X)) \ (mod\ R))$$

수식 1. TLSH 유사도 계산 공식

고 한다. 유사도 점수는 유사할수록 0에 가깝고, 유사하지 않을수록 큰 숫자의 결과를 출력한다. TLSH의 유사도 점수는 각 해시값의 헤더 간 거리와 바디 간 거리의 합으로 구하며, 이를 두 값 사이의 해밍 거리라 한다. TLSH 유사도 점수 계산은 (수식 1.)과 같다.

mod_diff(X,Y,R)는 크기가 R인 원형 큐에서 X, Y 사이의 최소 거리를 의미한다. X와 Y는 비교하고자 하는 TLSH 해시값이며, R은 해시값의 길이이다. 거리가 가까울수록 해시값은 서로 유사하다[9]. ssdeep 등 일부 LSH 알고리즘 사용 시 데이터 길이에 영향을 받아 유사도 비교 수행에 발생하는 노이즈를 줄이기 위하여 TLSH 유사도 비교를 사용한 클러스터링 연구가 진행된 바 있다[10].

Ⅲ. LSH 기반 유사도 측정 실험

LSH 기반 유사성 해시를 비교하기 위해 ssdeep, nilsimsa, TLSH 3가지 해시 알고리즘의 해시값 생성 시간, 펌웨어 파일 버전별 유사도 점수, 유사도 점수를 도출하는 시간을 측정하였다. 비교 대상은 Netis사 공유기 RX10 모델의 11개 버전 펌웨어 파일로 실험을 진행하였다. 유사도 점수는 RX10 모델의 최초 버전인 v.1.2.17381을 기준으로 그 이후의 버전들과 각각 비교하여 추출한 유사도 값이다. 해시값과 유사도 점수 생성 시간은 11개 버전의 펌웨어 생성 시간의 평균값을 나타내었다(표 1.).

해시값 생성 시간은 TLSH가 0.05초로 가장 빠르며, nilsimsa는 10.16초로 가장 많은 시간을 소요하였다. 유사도 점수 생성 시간은 ssdeep과 TLSH는 각각 0.000005초, 0.000003초이고, nilsimsa는 0.00003초를 소요하였다. 해시값과 유사도 점수 생성 시간 모두 TLSH가 가장 적게 소요되었고, 가장 많은 시간이 걸린 nilsimsa는 다른 해시와 약 10배의 차이를 보였다.

다음 (그림 1.)은 각 해시별 유사도 점수를 동일한 비율로 환산하여 도표로 나타내었다. nilsimsa 유사도 점수는 최대 22, 최소 4이고 TLSH 유사도 점수는 최대 246, 최소 210이며, ssdeep 유사도 점수는 모두 0이다. nilsimsa와 TLSH의 유사도 점수는 펌웨어의 버전별로 다른 점수 결과가 나타나지만, ssdeep 유사도 점수는 초기 버전 펌웨어와 나머지 펌웨어 버전들의 차이가 모두 0으로 같았다. 즉, ssdeep 알고리즘은 미세한 유사성의 차이를 판단할 수 없다는 것을 알 수 있다. ssdeep와 nilsimsa 알고리즘은 유사도 점수의 최소, 최댓값이 고정되어 있지만, TLSH는 고정되어 있지 않다. 그러므로 유사도 점수의 범위 제한이 없는 TLSH 알고리즘이 유사성의 미세한 정도를 잘 표현한다.

표 1. LSH 기반 유사성 해시(ssdeep, nilsimsa, TLSH) 특징

구분 \ hash		ssdeep	nilsimsa	TLSH
해시값 생성 시간 (s)		0.183116	10.16891	0.056712
Netis RX10 펌웨어 버전별 유사도 점수	v.1.6.99	0	16	237
	v.1.7.09	0	11	243
	v.1.7.14	0	4	237
	v.1.8.22	0	22	213
	v.1.8.23	0	19	210
	v.1.8.24	0	8	246
	v.1.8.25	0	12	246
	v.1.8.26	0	17	232
	v.1.8.31	0	10	217
	v.1.8.33	0	23	229
유사도 생성 시간 (s)		0.000005	0.00003	0.000003

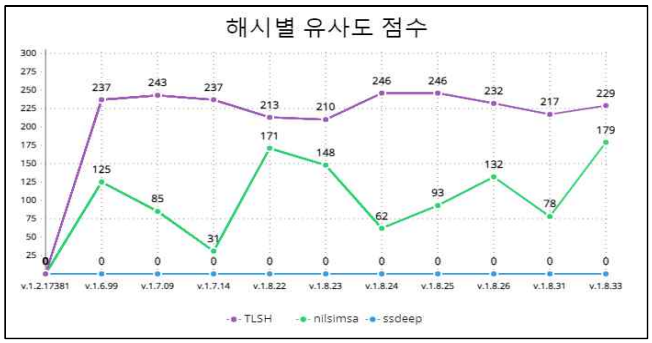

그림 1. 해시별 유사도 점수

Ⅳ. 결론

본 논문은 LSH와 LSH 기반 유사성 해시 3가지(ssdeep, nilsimsa, TLSH)에 대해 소개하고 성능을 비교·분석하였다. 해시값과 유사도 점수 생성 시간에서 nilsimsa가 가장 많은 시간을 소요하였고, 유사도를 평가하는 데에서는 ssdeep이 다른 LSH 기반 해시보다 세세한 차이를 판단하기에 부적절한 것으로 확인되었다. 즉, 해시값 및 유사도 점수 생성 시간과 유사도 평가에 있어 종합적으로 TLSH 알고리즘이 가장 효율적인 것을 알 수 있었다.

LSH 기반 유사성 해시는 모두 해시 충돌을 최대화한 해시로 유사할수록 유사한 해시값을 출력해주는 공통점이 있지만, 알고리즘별로 해시값 및 유사도 점수 생성 시간과 유사도 점수 판단에서는 성능의 차이가 존재하므로 필요에 따라 적절한 알고리즘을 선택해야 한다. LSH 알고리즘 중에서도 TLSH는 유사도 비교에 특히 좋은 성능을 보이므로, 이를 활용한 파일 중복 탐지, 악성파일 분류 등 파일 유사도 비교에 유용할 것으로 기대한다.

참 고 문 헌

[1] Shinho L., et al, “Dexofuzzy: Android malware similarity clustering method using opcode sequence,” Virus Bulletin, Nov. 2019.

[2] Moia V.H.G., Henriques M.A.A., “Similarity Digest Search: A Survey and Comparative Analysis of Strategies to Perform Known File Filtering Using Approximate Matching,” Security and communication networks, 2017.

[3] Slaney M., Casey M., “Locality-Sensitive Hashing for Finding Nearest Neighbors[lecture notes],”, IEEE Signal processing magazine 25(2), pp. 128-131, Mar. 2008.

[4] Shikhar Gupta, “Locality Sensitive Hashing,” Towards data science, Jun. 2018, (https://towardsdatascience.com/understanding-locality-sensitive-hashing-49f6d1f6134).

[5] Kornblum, J., “Identifying almost identical files using context triggered piecewise hashing.” Digital investigation 3, pp. 91-97, Jun. 2006.

[6] 박창욱, 정현지, 서광석, 이상진 “퍼지해시를 이용한 유사 악성코드 분류 모델에 관한 연구,” 정보보호학회논문지 22(6), pp. 1625-1336, Dec. 2012.

[7] Damiani E., et al, “An Open Digest-based Technique for Spam Detection,” ISCA PDCS 2004, pp. 559-564, 2004.

[8] 유정도, 김태규, 김인성, 김휘강, “멀티모달 기반 악성코드 유사도 계산 기법,” 정보보호학회논문지 Vol. 29 No. 2, pp. 347-363, Apr. 2019.

[9] Oliver, Jonathan, Chun Cheng, and Yanggui Chen. “TLSH--a locality sensitive hash,” In 2013 Fourth Cybercrime and Trustworthy Computing Workshop, pp. 7-13, Nov. 2013.

[10] 고동우, 김휘강, “API 콜 시퀀스와 Locality Sensitive Hashing을 이용한 악성코드 클러스터링 기법에 관한 연구,” 정보보호학회논문지 27(1), pp. 91-101, Feb. 2017.