SNS 환경에서 Apache Storm 기반 실시간 URL 수집 시스템

Apache Storm-based Real-time URL Collection System in SNS Environments

저자 (Authors)
남궁주홍, 길명선, 문양세
Juhong Namgung, Myeong-Seon Gil, Yang-Sae Moon

출처 (Source)
한국정보과학회 학술발표논문집 , 2019.6, 136-138(3 pages)

발행처 (Publisher)
한국정보과학회
The Korean Institute of Information Scientists and Engineers

URL
http://www.dbpia.co.kr/journal/articleDetail?nodeId=NODE08763105

APA Style
남궁주홍, 길명선, 문양세 (2019). SNS 환경에서 Apache Storm 기반 실시간 URL 수집 시스템. 한국정보과학회 학술발표논문집 , 136-138

SNS 환경에서 Apache Storm 기반 실시간 URL 수집 시스템

남궁주홍°, 길명선, 문양세
강원대학교 컴퓨터과학과
e-mail: {namgung_juhong, gils, ysmoon}@kangwon.ac.kr

Apache Storm-based Real-time URL Collection System in SNS Environments

Juhong Namgung°, Myeong-Seon Gil, and Yang-Sae Moon
Dept. of Computer Science, Kangwon National University

요 약

최근 SNS(Social Netowrk Service) 확산으로 대용량 데이터가 빠르게 생성됨에 따라 이를 위한 실시간 수집·처리 환경이 요구된다. 본 논문에서는 SNS 데이터에서 유효한 URL 데이터 수집을 위한 요구사항을 제안하고, 이를 만족하는 시스템을 실시간 분산 처리 프레임워크인 아파치 스톰(Apache Storm)에 구현한다. 그리고 실제 데이터를 수집하는 실험을 통해 제안 시스템의 기능과 성능을 평가한다. 특히, 단일 노드 시스템과 스톰 기반 분산 노드 시스템의 성능을 실험으로 비교한다. 실험 결과, 제안하는 시스템이 실시간 생성되는 SNS에서 정상적으로 유효한 URL 데이터를 빠르게 수집함을 확인하였고, 분산 노드 기반 시스템이 단일 노드보다 처리량을 최대 80배 증가, 지연시간을 1.7배 감소시킴을 보였다. 이러한 결과는 제안 시스템이 단일 노드 시스템보다 실시간 수집·처리 환경에 더 적합함을 의미한다.

1. 서 론

최근 인터넷과 스마트기기의 활성화로 소셜 네트워크 서비스(Social Network Service: SNS)의 사용이 급증하였다. 대표적인 SNS로는 트위터, 유튜브, 페이스북 등이 있다. 텍스트, 링크, 멀티미디어를 포함하는 SNS 데이터는 정치, 경제, 사회 분야에서 관계 분석, 여론·시장 분석, 트랜드 예측 등에 다양하게 활용된다[1]. 이러한 활용을 위해 복잡한 SNS 데이터에서 유효한 데이터를 수집하는 기술은 필수적인 요소이다. 특히, 실제 SNS 데이터는 매우 빠르게 생성되기 때문에 이를 위한 실시간 수집·처리 환경이 요구된다. 본 논문에서는 실시간으로 생성되는 SNS 데이터에서의 URL 수집에 초점을 맞춘다. SNS 사용자들은 URL을 통해 자신의 게시글과 관련된 추가적인 정보들을 공유한다. 그러나, 이러한 URL들은 애드웨어, 스파이웨어 등을 포함한 악성 URL인 경우도 많다. 본 연구에서는 URL을 기반으로 하는 다양한 행위 분석, 악성 URL 탐지 및 분석[2,3]에 활용 가능한 실시간 URL 수집 시스템을 제안한다.

SNS에서 유효한 URL을 수집하기 위해서는 다음과 같은 세 가지 요구사항을 만족해야 한다. 첫째, 복합적인 구조의 SNS 데이터를 분석하여 URL만을 분리할 수 있어야 한다. 둘째, 대부분의 SNS에서 사용되는 단축 URL을 확장하여 원본 URL을 파악할 수 있어야 한다. 셋째, 추출한 URL의 접속 상태 검증을 통해 죽은 링크(dead link)를 판별할 수 있어야 한다. 본 논문에서는 이러한 세 가지 기능을 통합 제공하는 URL 수집 시스템을 설계하고, 이를 실시간으로 처리하기 위해 분산 스트림 처리 프레임워크인 아파치 스톰(Apache Storm)[4,6]과 분산 메시징 큐인 아파치 카프카 (Apache Kafka)[5]를 기반으로 구현한다. 그리고, 제안하는 시스템의 성능을 평가하기 위해 트위터를 대상으로 단일 노드와 스톰 기반 분산 노드에서의 데이터 수집 실험을 진행한다. 실험을 통해, 스톰 기반 시스템이 단일 노드 대비 처리량을 최대 80배 향상, 지연시간을 최대 1.7배까지 감소시킴을 확인하였다. 이러한 결과로 보아, 제안 시스템은 대량의 URL 데이터가 요구되는 다양한 분석, 예측 환경에 적용 가능한 통합 URL 수집 환경이라 볼 수 있다.

2. 연구 배경

아파치 스톰[4, 6]은 분산 환경에서 실시간으로 스트림 데이터를 처리하는 대표적인 오픈소스 프레임워크이다. 스톰은 분산 클러스터에서 연산을 수행하며, 필요에 따라 구성된 노드를 확장할 수 있는 특징이 있다. 스톰은 토폴로지(Topology)라는 일련의 플랜(plan)으로 입출력과 처리를 정의하고, 동작시킨다. 토폴로지는 스파우트(Spout)와 볼트(Bolt)로 구성할 수 있다. 스파우트는 데이터 소스를 튜플로 바꾸어 스트림 데이터 형태로 볼트에 전달하는 역할을 한다. 그리고, 볼트는 받은 데이터를 처리하여 다음 볼트에 전달하거나, 처리 결과를 데이터베이스, 카프카 등 외부에 저장한다. 이러한 스톰의 중요한 특징은 병렬성이다. 같은 작업을 수행하는 스파우트 또는 볼트를 여러 개의 객체로 생성하면 서로 다른 서버에서 동시에 병렬로 동작시킬 수 있다.

아파치 카프카[5]는 분산 메시징 큐잉 시스템으로, 스톰과 같은 데이터 스트림 처리 프레임워크의 입력이나 파이프라인으로 활용된다. 카프카는 발행-구독 모델(publish-subscribe model)을 기반으

* 본 연구는 한국전력공사의 2018년 착수 에너지 거점대학 클러스터 사업(과제번호:R18XA05)과 2019년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. 2017R1A2B4008991)

로 동작하며, 실제 카프카가 동작하는 서버인 브로커(Broker), 브로커로 메시지를 발행하는 역할을 하는 프로듀서(Producer), 브로커로부터 메시지를 구독하는 역할을 하는 컨슈머(Consumer)로 구성되어 있다.

3. 스톰 기반 실시간 URL 수집 시스템

본 절에서는 SNS에서 유효한 URL을 수집하기 위한 요구사항을 먼저 제시한다. 그리고, 해당 요구사항을 만족하는 수집 시스템을 제안하고 이를 실시간 분산 처리를 위한 스톰 기반으로 설계 및 구현한다.

3.1 요구사항 및 해결방안

실시간으로 생성되는 SNS 데이터에서 유효한 URL을 수집하기 위해서는 다음 세 가지 요구사항이 필요하다. 첫째, 복합적인 구조의 SNS 데이터를 분석하여 URL만을 분리할 수 있어야 한다. 현재 서비스 중인 SNS에서는 각기 다른 형태로 정의된 데이터가 생성된다. URL 수집을 위해서는 먼저, 이러한 복합 구조의 SNS 데이터에서 수집하고자 하는 URL을 추출해야 한다. 본 논문에서 제안하는 시스템은 SNS 데이터를 문자열로 가정하고 URL의 정규 표현식을 지정하여 패턴 매칭(pattern matching)을 통해 URL을 추출한다. URL 추출을 위한 정규 표현식은 다음과 같다.

(https?|ftp|file)://[-a-zA-Z0-9+&@#/%?=~_|!:,.;]*[-a-zA-Z0-9+&@#/%=~_|]

둘째, 대부분의 SNS에서 사용되는 단축 URL을 확장하여 원본 URL을 파악할 수 있어야 한다. 단축 URL이란 길이가 긴 원본 URL을 짧은 URL로 대체하여 사용하는 방식으로, 짧은 URL이 긴 URL에 리다이렉션(redirection)되는 것을 의미한다[7]. 단축 URL은 글자 수에 제한이 있는 마이크로블로그(microblog)나 SMS에서 주로 사용된다. 대표적으로 트위터(http://t.co), 네이버(https://developers.naver.com/products/shortenurl/)에서 단축 URL을 생성해 주는 서비스를 제공한다. 하지만 단축 URL은 호스트 정보와 같은 실제 URL이 가지고 있는 정보를 숨기고 있다. 실제로 이를 악용하여 피싱이나 유해한 사이트 배포에 사용되기도 한다. 따라서, 정확한 URL 분석을 위해 단축 URL을 확장하여 실제 URL 주소를 얻는 과정이 필요하다. 본 논문에서 제안하는 시스템은 Java의 URLConnection 클래스를 사용하여 단축 URL을 검사하고, 이를 실제 URL로 확장한다.

셋째, 추출한 URL의 접속 상태 검증을 통해 죽은 링크를 판별할 수 있어야한다. 인터넷의 특성상 URL은 빠르게 생성 변경, 소멸된다. 따라서, 수집한 URL의 상태를 파악하여 죽은 링크 여부를 판단해야 한다. 죽은 링크 혹은 깨진 링크(broken link)는 영구적으로 이용할 수 없는 웹 페이지나 서버를 가리키는 링크를 말한다. 죽은 링크에 접속하면 일반적으로 웹 서버는 응답하지만 특정 페이지는 찾을 수 없을 때 나타나는 HTTP 404 오류를 볼 수 있다. 제안 시스템은 해당 URL의 HTTP 상태 코드를 확인하여 링크의 접속 유효성을 검사한다. 죽은 링크 검사를 통해 유효한 URL만을 수집함으로써 수집된 데이터에 대한 신뢰도를 높일 수 있다.

3.2 스톰 기반 실시간 SNS URL 수집 시스템

실시간 URL 수집을 위해 먼저, 제3.1절의 요구사항을 만족하는 시스템을 단일 노드 환경에서 구현한다. 그러나, 해당 시스템을 단일 노드에서 동작 시키면 특정 작업에 부하가 생기는 문제점이 있다. 그림 1을 보면, 카프카에서 데이터를 추출(consume)하여 URL을 저장(produce)하기까지의 전체 과정에서 URL을 확장(expand)하고 검증(validate)하는데 대부분의 시간을 사용하게 되는 것을 알 수 있다. 이는 특정 작업에 시간이 많이 소요되어 한 메시지가 처리될 때까지 다른 메시지가 대기하게 되는 것을 의미하고 이는 전체 성능의 저하를 가져오게 된다.

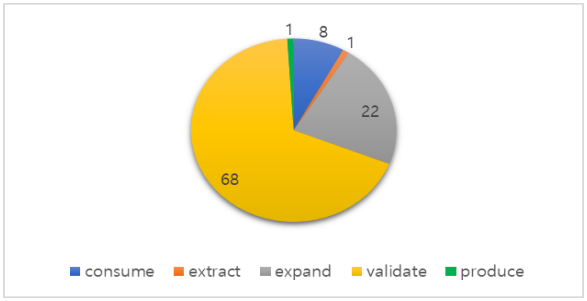

그림 1. URL 수집 과정 작업 당 소요 시간 비율(%).

따라서, 본 논문에서는 이를 해결하기 위해 분산 스트림 처리 프레임워크인 아파치 스톰을 사용하여 URL 수집 시스템을 구현한다. 그림 2는 실시간 URL 수집 분산 처리 시스템의 전체 구조도이다. 시스템은 SNS 데이터에서 유효한 URL 데이터를 저장하는 작업을 하는 스톰과 SNS와 URL 데이터를 저장하는 카프카로 구성된다. 시스템의 동작 구조는 다음과 같다. 먼저, SNS 데이터를 카프카로 저장한다. 그리고, KafkaSpout는 카프카에 저장된 데이터를 읽어 ExtractBolt에게 넘긴다. ExtractBolt는 SNS 데이터에서 정규 표현식과 패턴 매칭을 이용하여 URL을 추출해낸 뒤 다음 볼트인 ExpandBolt로 전송한다. ExpandBolt는 단축 URL을 확장한 결과를 ValidateBolt로 전송한다. ValidateBolt는 HTTP 상태코드를 확인하여 유효한 URL만 다음 볼트로 전송한다. 마지막으로, KafkaBolt는 이전 볼트에서 전송된 유효한 URL을 카프카에 저장한다. 위 과정을 통해 실시간으로 SNS 데이터에서 유효한 URL을 수집할 수 있다.

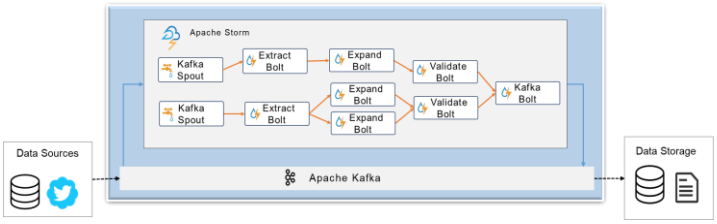

그림 2. 실시간 URL 수집 분산 처리 시스템 구조도.

제안 시스템은 스톰의 병렬성을 활용하여 처리 성능을 향상시키고자 한다. 즉, 부하가 생기는 작업을 여러 개의 객체인 볼트로 동작시켜 전체 처리량을 향상시키고자 한다. 그림 1의 실험 결과를 고려하여 주요 병목인 ExpandBolt와 ValidateBolt의 수를 증가시켜 성능을 향상시킬 수 있다. 또한, 시스템에서 카프카를 사용한 이유는 특정 SNS에 제한되지 않고 다양한 종류의 SNS 데이터를 처리하여 URL을 수집·저장할 수 있게 하기 위함이다.

4. 실험 결과

본 절에서는 제안 시스템을 사용하여 실제 URL을 수집한 결과를 설명하고 이에 대한 성능을 평가한다. 실험에 사용한 스톰 클러스터 서버의 하드웨어 사양은 마스터 노드(Intel Xeon E5-2630 2.40Hz 8 Core CPU, 32GB RAM) 1대와 슬레이브 노드(Intel Xeon E5-2630 2.40Hz 6 Core CPU, 32GB RAM) 8대이다. 소프트웨어 환경으로, 스톰은 2.0.0, 카프카는 2.11 버전을 사용하였다.

4.1 URL 수집 결과

본 절에서는 제안하는 시스템을 통해 대표적 SNS인 트위터에서 URL을 수집한 결과를 확인한다. 먼저 트위터에서 제공하는 API를 사용하여 실시간으로 카프카에 데이터를 전송한다. 시스템은 카프카에서 데이터를 읽어와 URL을 수집하는 과정을 거쳐 다시 카프카로 데이터를 저장한다. 그림 3과 4는 각각 URL 수집 과정 중 단축 URL을 확장한 결과와 죽은 링크 검사 결과를 나타낸다. 그림 3에서 볼 수 있듯이, SNS에서 많이 발생되는 단축 URL이 원본 URL로 확장된 것을 확인할 수 있다. 또한, 그림 4와 같이 HTTP 코드가 400 이상인 유효하지 않은 URL은 수집하지 않음을 확인하였다.

그림 3. 단축 URL의 확장 결과.

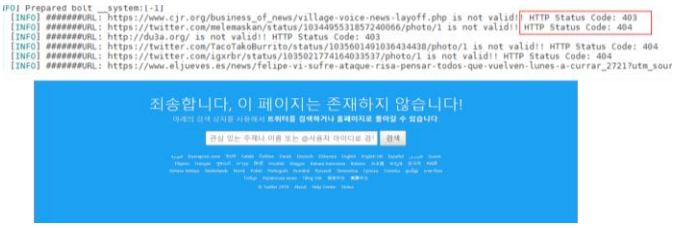

그림 4. 죽은 링크 검사 결과.

마지막으로, 그림 5의 결과를 통해 유효한 URL만이 카프카로 수집됨을 확인한다. 위 실험을 통해 제안하는 시스템이 실시간으로 생성되는 트위터에서 정상적으로 URL을 추출하고, 단축 URL을 확장하며 죽은 링크를 검사하여 유효한 URL만을 수집하는 것을 확인하였다.

```
https://twitter.com/iu_bluewing/status/1035480326141706240/photo/1
http://bestfanarmy.com/Winners
https://twitter.com/volks_dd/status/1035097576695971840/photo/1
https://twitter.com/realdonaldtrump/status/1035650292782653441
https://twitter.com/rodrigopleasee/status/1033233655206952960/photo/1
https://twitter.com/fatezeyo62/status/1033608826132430849/photo/1
https://twitter.com/Metalmike3/status/1035209122797342720/photo/1
https://twitter.com/i/web/status/1035734488188428289
https://twitter.com/_S_K_B/status/1034641999981211648/photo/1
https://twitter.com/gozz_sss/status/1035642382409129984/photo/1
^CProcessed a total of 158 messages
[team1@SN07 kafka]$
```

그림 5. 최종 URL 수집 결과.

4.2 성능 비교

본 절에서는 URL 수집 시스템을 단일 노드에서 구현한 시스템과 스톰 기반 분산 노드에서 구현한 시스템의 성능을 비교하는 실험으로 비교한다. 그림 6은 단일 노드와 스톰 기반 분산 노드의 처리량 결과이다. 그림에서 노드 수 '1'은 단일 노드를 의미하고 노드 수 '5~8'은 스톰 기반 분산 노드에서 슬레이브 숫자를 증가한 것을 의미한다. 그림에서 볼 수 있듯이, 단일 노드 대비 스톰 기반 분산 노드 기반 시스템에서 처리량이 크게 증가한 것을 확인할 수 있다. 또한, 스톰 기반 분산 노드에서 노드 수가 증가함에 따라 처리량이 증가하는 것을 알 수 있다.

그림 7은 단일 노드와 스톰 기반 분산 노드 시스템의 지연시간 결과이다. 처리량과 마찬가지로 분산 노드 시스템에서 단일 노드 대비 지연시간이 크게 낮아진 것을 확인할 수 있다. 노드 수 '5~8'에서 큰 변화가 없는 이유는 지연시간이 노드 개수에 비례하지는 않기 때문이다. 두 실험 결과를 요약하면, 스톰 기반 분산 노드 시스템을 통해 단일 노드 대비 처리량을 최대 80배, 지연시간을 최대 1.7배 향상시킬 수 있다. 이는 분산 노드 시스템이 단일 노드 시스템에 비해 실시간 환경에 더욱 적합함을 의미한다.

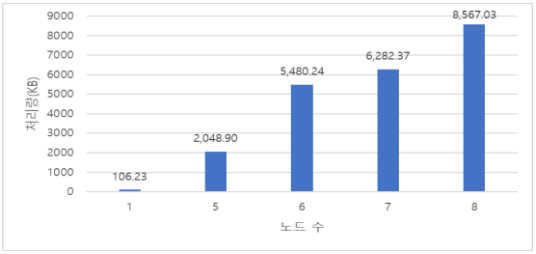

그림 6. 노드 수에 따른 처리량 그래프.

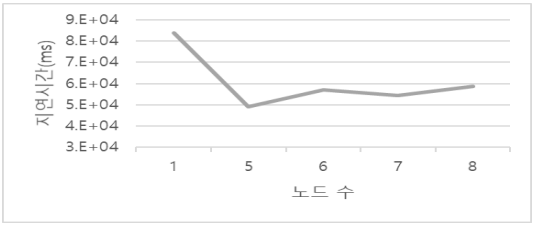

그림 7. 노드 수에 따른 지연시간 그래프.

5. 결론

본 논문에서는 실시간으로 생성되는 SNS에서 유효한 URL을 수집하기 위한 요구사항을 제안하고, 이를 만족하는 시스템을 구현하였다. 단일 노드 기반 URL 수집 시스템은 특정 작업에 부하가 생겨 실시간 환경에서 성능이 저하되는 문제가 있다. 본 논문에서는 이를 분산 처리 프레임워크인 스톰에서 설계 및 구현하여, 부하가 생기는 작업을 더 많은 노드가 처리할 수 있게 하였다. 실제 데이터를 실시간으로 수집하는 실험 결과, URL 수집을 위한 요구사항을 만족하면서 정상적으로 유효한 URL 데이터가 수집됨을 확인하였다. 또한, 스톰 기반 분산 시스템과 단일 노드 시스템의 성능을 비교한 결과, 분산 노드 시스템이 단일 노드 시스템보다 처리량을 최대 80배 향상, 지연시간을 1.7배 감소시켰다. 향후 연구로는 URL 수집 단계별 최적화 및 부하 문제 해결을 통해 전체 시스템의 성능을 향상시키는 연구를 수행할 예정이다.

참고 문헌

[1] 김동완, "빅데이터의 분야별 활용사례," *경영논총*, 제34권, pp. 39-52, 2013년 12월.

[2] 원대연, 박기정, 박영준, 심규배, 이재웅, 김용혁,"네이브 베이지안 알고리즘과 URL 분석에 기반을 둔 스팸 트윗 필터링," *한국정보과학회 2011 학술발표논문집*, 제38권, 제2호(B) pp. 375-378, 2011년 11월.

[3] R. Verma and A. Das, "What's in a URL: Fast Feature Extraction and Malicious URL Detection," In *Proc. of the 3rd ACM on Int'l Workshop on Security and Privacy Analytics*, Scottsdale, Arizona, pp. 55-63, Mar. 2017.

[4] Apache Storm, http://storm.apache.org/.

[5] Apache Kafka https://kafka.apache.org/.

[6] P. Goetz and B. O'Neill, Storm Blueprints: Patterns for Distributed Real-time Computation, Packt Publishing Ltd., 2014.

[7] 윤수진, 박정은, 최창국, 김승주, "SHRT: 유사 단어를 활용한 URL 단축 기법", *한국통신학회논문지*, 제38권, 제6호, pp. 473-484, 2013.