9장 웹크롤러설계

웹 크롤러는 검색엔진에서 널리쓰이는 기술로 웹에 새롭게 올라오거나 갱신된 콘텐츠를 찾아내는 것이 주된 목적

- 검색 엔진 인덱싱 - 크롤러의 가장 보편적인 용례
- 웹 아카이빙 - 나중에 사용할 목적으로 장기보관하기 위해 웹에서 정보를 모으는 절차
- 웹 마이닝 - 웹에서 데이터를 마이닝하는 것

1단계

기본 알고리즘 : URL 집합이 입력으로 주어지면 웹페이지 다운로드 → 다운 받은 페이지에서 다시 URL 추출 → 다시 위 과정 반복

요구사항 추정하기

- 용도 : 검색 엔진 인덱싱
- 용량 : 매달 10억개의 웹 페이지를 수집해야함
- 저장 기간 : 5년
- 중복 콘텐츠 : 무시

만족시켜야할 속성

- 규모 확장성 : 병행성을 활용하면 효과적으로 크롤링 가능
- 안정성 : 잘못 작성된 html, 잘못된 응답 등 유연하게 대응해야 함
- 예절 : 수집 대상의 웹사이트에 너무 많이 보내지 않기
- 확장성 : 새로운 형태의 콘텐츠를 지원하기가 쉬어야 함

규모 측정

- 평균 처리량 : 10억/달 → 400페이지/초
- 최대 피크 : 2 * 평균 처리량 = 800페이지/초
- 웹 페이지의 평균 크기 500kb

- 10억 페이지 * 500kB = 500TB/월
- 5년간 저장 = 60month * 500TB = 30PB

2단계 - 개략적인 설계안 제시

1. 시작 URL 집합 : 크롤링의 시작점, 일반적으로는 도메인 네임 자체를 쓸 수 있으나 도메인 별로 카테고리를 분류하고, 유명한 사이트들을 분류하여 집합을 만들기도 한다.
2. 미수집 URL 추출 : 새로운 URL를 추출한다. - 일반적으론 FIFO. queue 사용한다.
3. HTML 다운로더 → DNS : DNS를 통해 HTML를 다운로드 한다.
4. 컨텐츠 파서 : 콘텐츠 파서 컴포넌트는 다운로더에 영향을 끼칠 수 있으므로 별도 컴포넌트로 작성한다.
5. 중복 콘텐츠 ← 컨텐츠 저장소 - 디스크와 메모리를 둘다 사용하는 저장소를 골라서 인기있는 콘텐츠는 캐시를 탈 수 있도록 한다.
6. URL 추출기 - 상대 경로, 절대 경로를 기반으로 URL을 절대 경로로 변환하여 추출한다.
7. URL 필터 - 특정 콘텐츠 타입이나 파일 확장자가 같은 URL는 제외한다.
8. 이미 방문한 URL → URL 저장 : 블룸필터나 해시테이블 등을 이용할 수 있다.

   → 미수집 URL로

3단계 - 상세 설계

DFS vs BFS

- DFS의 경우 그다지 좋은 선택이 아닐 가능성이 있다. 그래프의 크기가 클 경우 어느정도 깊숙히 가게될지 가늠하기 어렵다 /
- BFS의 문제점
  - 한페이지에서 나오는 링크의 상당수는 같은 서버로 되돌아간다. 큐에 한 사이트의 여러 링크가 연속적으로 쌓이면서 예의 없는 사이트로 간주될 수 있다.
    - 🤫 : DFS도 동일하지 않나?
  - 기본적으로 URL간의 우선순위를 두지 않아서 중요하지 않은 페이지도 다룰 수 있다.

미수집 URL 저장소

- 예의 바른 크롤러를 위해 지켜야하는 원칙 : 동일 웹사이트에 대해서 한번에 한 페이지만 요청할 것
    - 각 다운로드 스레드 별로 별도의 FIFO 큐를 가지고 있어서 해당 큐에서 꺼낸 URL만 다운로드 한다.

```
flowchart

QR[Queue Router]-->MP[Mapping Table]
QR-->B1[Queue #1]
QR-->B2[Queue #2]
QR-->B3[Queue #3]
B1-->QS[Queue Selector]
B2-->QS
B3-->QS
QS-->WT1[Worker Thread #1]
QS-->WT2[Worker Thread #2]
QS-->WT3[Worker Thread #3]
```

    - Queue Router : Mapping table에서 호스트와 큐간의 관계를 참고하여 입력된 URL을 적절한 큐로 전송
    - Queue Selector : 큐들을 순회하면서 URL을 꺼내어 적절하게 작업 스레드로 전달
- 우선순위
    - URL의 우선순위를 나눌떄에는 Page Rank 트래픽양, 갱신 빈도등을 고려해서 Prioritizer를 구현한다.
    - 이전의 다이어그램에서 호스트당 하나의 큐가 아니라 우선순위 당 하나의 큐로 설정하고 queue router에서 적절하게 우선순위별로 배분한다고 보면 된다.
- 우선순위를 지키는 예의바른 큐 : 우선순위 큐를 먼저두고 예의 바른 큐 배분하는 장치를 후면에 배치면 두가지 요건을 모두 지킬 수 있다.
- 신선도 : 웹콘텐츠는 계속해서 변경될 수 있기 때문에 재수집할 필요성이 있다. 이를 최적화하기 위해 페이지 변경이력과 우선순위를 활용하여 중요 페이지부터 더 자주 재수집할 수 있다.
- 미수집 저장소를 위한 지속성 저장장치 : 거의 대부분 디스크에 디록하지만, 메모리 버퍼를 사용하여 IO비용을 줄일 수 있다
- http 다운로더
    - 로봇 제외 프로토콜 : robots.txt - 특정 path에 콘텐츠는 수집을 허용하지 않는다는 내용 등을 담을 수 있다

- 성능 최적화
  - 분산 크롤링 - 여러 서버에서 분산하여 크롤링을 하는 방식
  - 도메인 이름 변환 결과 캐시 : DNS 는 크롤러의 대표적인 병목중 하나, DNS 결과를 캐시해둬 질의과정을 없앨 수 있다
  - 지역성 : 지역별로 크롤링 노드를 분산하여 다운로드, 응답시간을 줄일 수 있다
  - 짧은 타임아웃: 대기시간을 설정해 응답이 없다면 바로 다음 페이지를 크롤링 한다
- 안정성
  - 안정해시 - 부하를 분산할 때 적용가능하다. 이를 사용하면 다운로더 서버를 쉽게 추가하고 삭제가 가능하다.
  - 크롤링 상태 수집 및 데이터 저장 : 지속적으로 수집된 데이터를 저장장치에 기록하자.
  - 예외처리 : 예외가 발생해도 전체 시스템이 중단되면 안된다.
- 확장성 : 진화하지 않는 시스템이라는 것을 보장할 수 없으므로, 새로운 형테의 콘텐츠를 쉽게 지원할 수 있도록 신경써야한다.
- 문제있는 콘텐츠 감지 및 회피
  - 중복 콘텐츠 : 웹콘텐츠의 30프로는 중복이다. 해시나 체크섬을 활용할 것
  - 거미덫 : 무한 루프에 빠뜨리도록 설계한 웹페이지 → 최대 길이 제한
  - 데이터 노이즈 : 어떤 콘텐츠는 거의 가치가 없음

마무리

추가로 논의해볼 법한 내용

- SSR(Server side rendering) : client 코드로 동적으로 콘텐츠를 불러오는 경우 서버 내에서 동적 렌더링을 적용하여 파싱한다. (헤드리스 크롬 등)
- 원치 않는 페이지 필터링 : 저장 공간 등 크롤링에 소요되는 자원이 유한하기 때문에 스팸 방지 컴포넌트를 두어 해소한다.
- 데이터베이스 다중화 및 샤딩 : replication, sharding과 같은 기법을 사용하여 data layer의 가용성 규모 확장성 안정성을 향상한다.
- 수평적 규모 확장성 : 크롤링을 위해 수평적으로 규모가 확장할 수 있도록 무상태 서버로 만들어라.
- 가용성, 일관성, 안정성

- 데이터 분석 솔루션