9️⃣

# 9장. 웹 크롤러 설계

**크롤러가 사용되는 곳**

- 검색 엔진 인덱싱
- 웹 아카이빙
- 웹 마이닝
- 웹 모니터링

→ 웹 크롤러는 데이터의 규모에 따라 달라지므로 설계할 웹 크롤러가 감당해야 하는 데이터의 규모와 기능을 알아내야 한다.

## 1. 문제 이해 및 설계 범위 확정

**웹 크롤러의 기본 알고리즘**

1. URL 집합이 입력으로 주어지면, 해당 URL들이 가리키는 모든 웹 페이지를 다운로드한다.
2. 다운받은 웹 페이지에서 URL들을 추출한다.
3. 추출된 URL들을 다운로드할 URL 목록에 추가하고 위 과정을 처음부터 반복한다.

**웹 크롤러를 위해 고려해야 하는 요구사항**

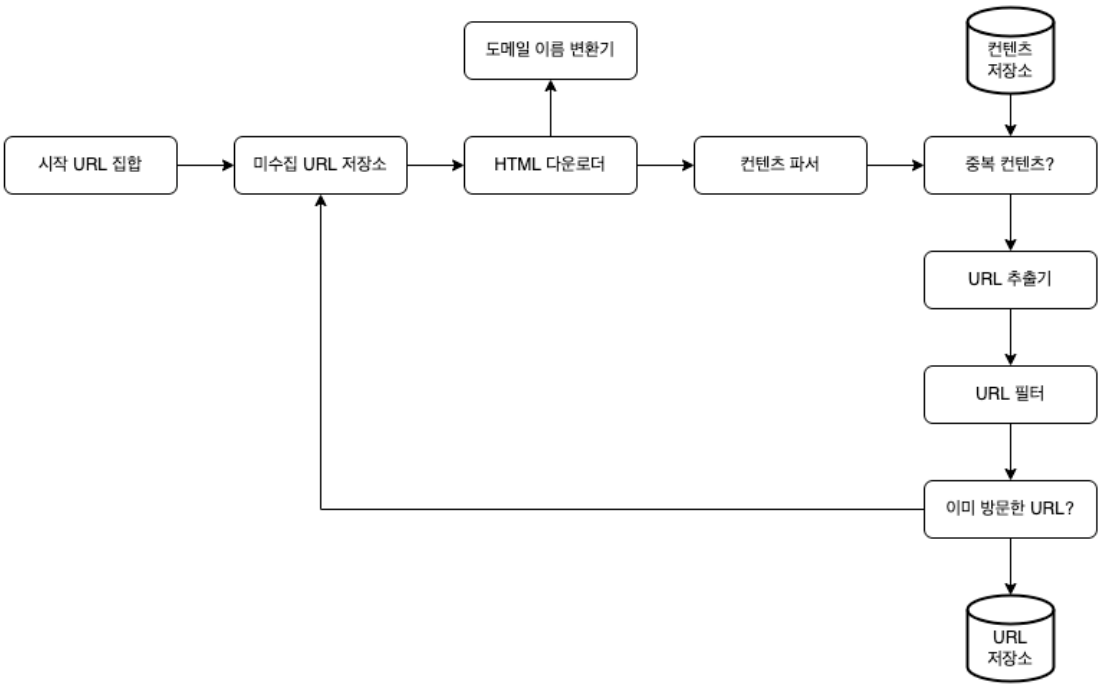
- 용도 (검색 엔진 인덱스 생성용, 데이터 마이닝, 등등..)
- 매달 수집해야하는 웹 페이지 규모, 예)약 10억개
- 수정된 웹 페이지도 고려되어야 하는지? (크롤링 후 수집된 결과 다시 가져오기)
- 웹 페이지 저장 기간, 예) 약 5년
- 중복된 콘텐츠는 무시한다.

등등..

**웹 크롤러가 만족시켜야하는 속성**

- 규모 확장성 : 웹에는 수십억개의 페이지가 존재하는 만큼 방대하다. ⇒ 병행성이 필요
- 안정성(robustness) : 크롤러는 비정상적인 입력이나 환경에 잘 대응해야 한다.
- 예절(politeness) : 수집 대상 사이트에 짧은 시간동안 너무 많은 요청을 보내면 안된다.
- 확장성(extensibility) : 새로운 형태의 콘텐츠를 지원하기 쉬워야한다.

## 2. 개략적 설계안 제시 및 동의 구하기




**시작 URL 집합**

전체 웹을 크롤링해야하는 경우 시작 URL을 고를 때

- 크롤러가 가능한 많은 링크를 탐색할 수 있도록 한다.

  → 주로 전체 URL 공간을 작은 부분집합으로 나누는 전략 사용

→ 주제별로 다른 시작 URL을 사용하는 것

미수집 URL 저장소

다운로드할 URL을 저장 광리하는 컨포넌트 (FIFO 큐라고 생각하면 됨)

HTML 다운로더

인터넷이서 웹 페이지를 다운로드하는 컴포넌트

도메인 이름 변환기

URL을 IP 주소로 변환해주는 절차

콘텐츠 파서

웹 페이지를 다운로드하면 파싱과 검증을 거쳐야합니다.

중복 콘텐츠인가?

웹 페이지 해시 값을 비교하여 중복 컨텐츠를 줄일  수 있다

콘텐츠 저장소

HTML 문서를 보관하는 시스템으로 저장할 데이터의 유형, 크기, 저장소 접근 빈도, 데이터의 유효 기간등을 종합적으로 고려해야 한다.

- 데이터의 양이 너무 많으므로 대부분의 콘텐츠는 디스크에 저장합니다.
- 인기 있는 콘텐츠는 메모리에 두어 접근 지연시간을 줄일 수 있습니다.

URL 추출기

URL 추출키는 HTMl 페이지를 파싱하여 링크들을 골라내는 역할을 합니다.

URL 필터

URL 필터는 특정한 콘텐츠 타입이나 파일 확장자를 갖는 URL 접속 시 오류가 발생하는 URL, 접근 제외 목록에 포함된 URL 등을 크롤링 대상에서 배제하는 역할을 합니다.

이미 방문한 URL인가?

이미 방문한 URL이나 미수집 URL 저장소에 보관된 URL을 추적할 수 있습니다.

블롬 필터(bloom filter)나 해시 테이블이 널리 쓰입니다.

URL 저장소

이미 방문한 URL을 보관하는 장소입니다.

**웹 크롤러 작업 흐름**

1. 시작 URL들을 미수집 URL 저장소에 저장한다.

2. HTML 다운로더는 미수집 URL 저장소에서 URL 목록을 가져온다.

3. HTML 다운로더는 도메인 이름 변환기를 사용하여 URL의 IP 주소를 알아내고, 해당 IP 주소로 접속하여 웹 페이지를 다운받는다.

4. 콘텐츠 파서는 다운된 HTML 페이지를 파싱하여 올바른 형식을 갖춘 페이지인지 검증한다.

5. 콘텐츠 파싱과 검증이 끝나면 중복 콘텐츠인지 확인한느 절차를 개시한다.

6. 중복 콘텐츠인지 확인하기 위해서, 해당 페이지가 이미 저장소에 있는지 본다.

- 이미 저장소에 있는 콘텐츠인 경우에는 처리하지 않고 버린다.
- 저장소에 없는 콘텐츠인 경우에는 저장소에 저장한 뒤 URL 추출기로 전달한다.

7. URL 추출기는 해당 HTML 페이지에서 링크를 골라낸다.

8. 골라낸 링크를 URL 필터로 전달한다.

9. 필터링이 끝나고 남은 URL만 중복 URL 판별 단계로 전달한다.

10. 이미 처리한 URL인지 확인하기 위하여, URL 저장소에 보관된 URL인지 살핀다. 이미 저장소에 있는 URL은 버린다.

11. 저장소에 없는 URL은 URL 저장소에 저장할 뿐 아니라 미수집 URL 저장소에도 전달한다

## 3. 상세설계

**DFS를 쓸 것인가, BFS를 쓸 것인가**

- 웹은 directed graph와 같습니다.
- 페이지는 노드, URL은 엣지라고 보면 좋습니다.
- 크롤링에서 DFS를 쓸 경우, 그래프 크기가 클 경우, 어느 정도로 깊숙이 가게 될지 가늠이 되지 않습니다.
- 따라서 웹 크롤러는 보통 BFS를 사용합니다.
- 그러나 구현법에는 두 가지 문제점이 있습니다.
    - 한 페이지에서 나오는 링크의 상당수는 같은 서버로 되돌아 갑니다.
    - 표준적 BFS 알고리즘은 우선순위를 두지 않습니다.

**미수집 URL 저장소**

- 미수집 URL을 통해 위의 두가지 문제를 해결할 수 있다.

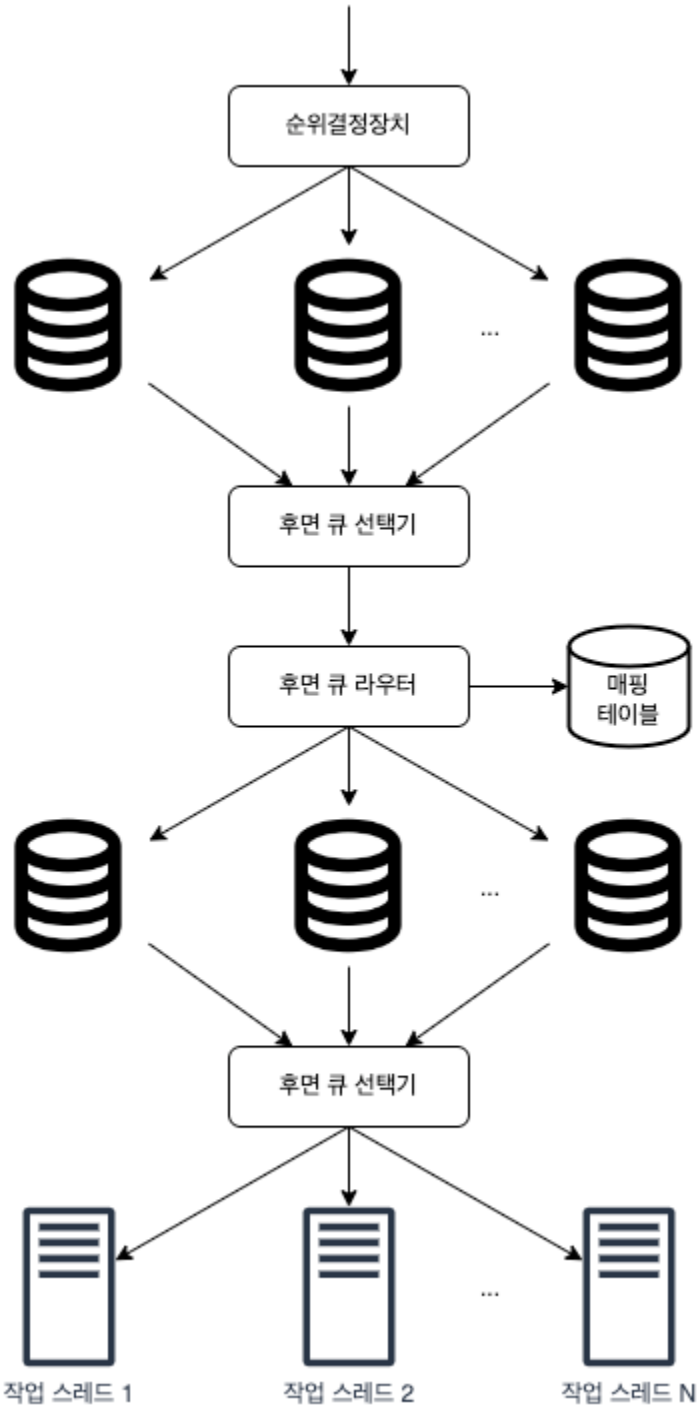
  예의

  - 웹 크롤러는 수집 대상 서버로 짧은 시간 안에 많은 요청을 보내는 것을 삼가해야합니다.

  우선순위

  - URL 마다 페이지 랭킹을 매길 수 있습니다. (트래픽 양이나 갱신 빈도 등)

위의 예의와 우선순위를 고려한 설계는 다음과 같습니다.




전면 큐(front queue) : 우선순위 결정 과정을 처리

후면 큐(back queue) : 크롤러가 예의 바르게 동작하도록 보증

**신선도**

웹 페이지는 수시로 추가되고 삭제, 변경됩니다. → 데이터의 신선함을 유지하기 위해서는 주기적으로 업데이트를 해주어야 합니다.

**최적화 전략**

- 웹 페이지의 변경 이력(update hisotry) 활용
- 우선순위를 활용하여, 중요한 페이지는 좀 더 자주 재수집

**미수집 URL 저장소를 위한 지속성 저장장치**

URL의 수는 수억 개에 다라므로 모두 메모리에 보관은 적합하지 않습니다.

## HTTP 다운로더

## Robots.txt

웹사이트가 크롤러와 소통하는 표준적인 방법

## 성능 최적화

- 분산 크롤링
  - 성능을 높이기 위해 크롤링 작업을 여러 서버에 분산합니다.
- 도메인 이름 변환 결과 캐시
  - DNS Resolver는 크롤러 성능의 병목 중 하나입니다.
  - DNS 조회 결과로 얻어진 도메인 이름과 IP 주소 사이의 관계를 캐시에 보관해 놓고 크론 잡 등을 돌려 주기적으로 갱신하도록 하면 성능을 효과적으로 높일 수 있습니다.
- 지역성
  - 크롤링 작업을 수행하는 서버를 지역별로 분산하는 방법
- 짧은 타임아웃
  - 어떤 웹 서버는 응답이 느리거나 아예 응답하지 않으므로, 최대 기다릴 시간을 정해놓습니다.

### 안정성

안정 해시(consistent hashing) : 다운로드 서버들에 부하를 분산할 때 적용 가능한 기술입니다.

크롤링 상태 및 수집 데이터 저장 : 장애가 발생한 경우에도 쉽게 복구할 수 있도록 크롤링 상태와 수집된 데이터를 지속적 저장장치에 기록해 두는 것이 바람직합니다.

예외 처리(exception handling) : 대규모 시스템에서 에러(error)는 불가피할 뿐 아니라 흔하게 벌어지는 일입니다.

데이터 검증(data validation) : 시스템 오류를 방지하기 위한 중요 수단 가운데 하나입니다.

### 확장성

위 첫번째 그림에서 `URL 추출기` 자리에 `PNG 다운로더`, `웹 모니터` 등의 모듈과 함께 확장 모듈로 설계가 가능합니다.

### 문제 있는 콘텐츠 감지 및 회피 전략#

- 중복 콘텐츠
  - 웹 컨텐츠의 30% 가량은 중복입니다.
- 거미 덫
  - 크롤러를 무한 루프에 빠뜨리도록 설계한 웹 페이지입니다.
  - 최대 길이를 제한함으로 회피할 수 있습니다.
  - 이런 덫의 경우, 기익할 정도로 웹 페이지가 많은 게 일반적입니다.
- 데이터 노이즈
  - 어떤 콘텐츠는 가치가 없습니다.

## 4단계 마무리

아래의 사항을 추가적인 이야기를 해볼 수 있습니다.

- **서버 측 랜더링(SSR)**: 많은 사이트가 자바 스크립트, AJAX 등의 기술을 사용해서 링크를 즉석으로 만들어 냅니다.

- **원치 않는 페이지 필터링**: 저장 공간 등 크롤링에 소요되는 자원은 유한하기 때문에 스팸방지 컴포넌트르 두어 품질이 조악하거나 스팸 성인 페이지를 걸러내도록 해 둡니다.
- **데이터베이스 다중화 및 샤딩**: 다중화나 샤딩 같은 기법 적용시, 데이터 계층의 가용성,규모 확장성, 안정성이 향상됩니다.
- **수평적 규모 확장성 :** 서버가 상태정보를 유지하지 않도록 하는 것, 즉 무상태(stateless)서버로 만드는 것입니다.
- **가용성, 일관성, 안정성**: 필수적으로 고려할 사항입니다.
- **데이터 분석 솔루션(analytics)**: 데이터를 수집하고 분석하는 것은 어느 시스템에게나 중요합니다.