서버리스 플랫폼에서 연속된 콜드 스타트 완화를 위한 Pre-Warming 기법

김세진, 유문상, 유헌창
고려대학교 컴퓨터학과
sejjj120@korea.ac.kr, msyhu@korea.ac.kr, yuhc@korea.ac.kr

Mitigating Cold Start Chain by Pre-Warming Containers in Serverless Platform

Sejin Kim, Moonsang Yhu, Heonchang Yu
Dept. of Computer Science and Engineering, Korea University

요 약

최근 인프라를 관리할 필요가 없고 폭발적으로 늘어나는 요청을 유연하게 대처할 수 있는 장점 때문에 서버리스 컴퓨팅 사용이 늘어나고 있다. 하지만 서버리스 컴퓨팅은 사용자 코드의 실행 환경을 준비하기 위한 콜드 스타트 과정이 필요하고, 서비스가 복잡해짐에 따라 전체 실행 시간 중 콜드 스타트로 인한 지연시간이 늘어나는 문제가 발생한다. 본 논문에서는 서버리스 컴퓨팅 기반의 워크플로우에 대해 콜드 스타트로 인한 지연 시간을 완화하는 아키텍처 및 기법을 제안한다.

1. 서론

최근 컴퓨팅 인프라를 관리할 필요 없이 비즈니스 로직을 담당하는 코드만 작성하여 배포하는 서버리스 컴퓨팅이 떠오르고 있다.[1] 대표적인 서버리스 컴퓨팅 서비스인 AWS Lambda 를 사용하면 개발자는 함수 단위로 코드를 작성하여 배포하고 해당 함수가 호출되는 횟수만큼만 비용을 지불하게 된다. 이외에도 컨테이너를 통해 서비스를 배포할 수 있다. 이러한 서버리스 컴퓨팅은 컴퓨팅 분야의 새로운 패러다임으로 불리지만, 몇가지 문제점이 존재한다. 그 중 콜드 스타트 문제는 다양한 관점에서 많은 연구가 진행중이다.

콜드 스타트 문제란, 개발자가 배포한 서버리스 형태의 서비스를 실행 시키기 위한 환경이 만들어질 때까지의 지연 시간이 발생하는 문제를 말한다. 서비스를 제공하기 위한 환경을 만드는 이유는 서버리스 플랫폼이 자원 활용률을 최대한 높이는 데에 초점을 맞추기 때문이다. 만약 배포된 서비스가 일정 시간 이상 호출되지 않는다면 서버리스 플랫폼은 해당 서비스에 할당된 자원을 회수하고, 다시 요청이 들어올 때 실행 환경을 만든 다음 함수를 실행한다.

서비스가 복잡해짐에 따라 2 개 이상의 마이크로서비스를 사용해 워크플로우를 구성하는 경우가 증가하고 있다.[2] 본 논문은 서버리스 플랫폼에서 일련의 워크플로우로 연결된 서비스가 호출되는 상황에서 발생할 수 있는 콜드 스타트를 완화하기 위한 아키텍처를 제안하고 실험을 통해 연속된 콜드 스타트가 완화되어 전체 실행 시간이 감소되는 것을 확인하고자 한다.

2. Kubernetes 기반 서버리스 플랫폼

A. Kubernetes

Kubernetes 는 구글에서 관리하며 컨테이너 형태의 애플리케이션을 배포, 스케일링 및 관리하는 컨테이너 오케스트레이션 시스템이다.[3] 독립된 환경을 제공하는 컨테이너를 관리하기 위한 도구 중 사실상 표준(De facto)으로 자리잡고 있다.

B. Knative

Knative 는 Kubernetes 환경에서 서버리스 형태의 서비스를 제공하는 오픈소스 프로젝트이며 Serving 컴포넌트와 Eventing 컴포넌트로 이루어져 있다.[4] Serving 컴포넌트는 컨테이너 단위로 배포된 코드를 서버리스 형태로 제공 및 관리하는 역할을, Eventing 컴포넌트는 해당 코드를 호출할 수 있는 트리거를 담당한다. 개발자가 웹서버 형태로 서비스를 작성하고 배포하면, Knative 에서 제공하는 URL 로 각 서비스에 접근이 가능해진다.

C. 콜드 스타트

서버리스 플랫폼의 특징 중 하나는 유연한 자원 관리이다. Knative 또한 함수가 일정 시간 호출되지 않으

면 해당 서비스의 컨테이너를 종료 시키는 방식으로 낭비되는 자원을 최대한 줄이고, 서비스가 다시 호출될 때 함수 실행을 위한 컨테이너를 다시 생성한다. 서버리스 플랫폼에서 서비스가 제공되기 까지의 과정은 다음과 같이 나타낼 수 있다.

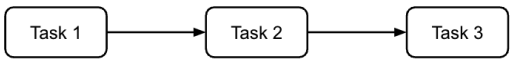
1. 컨테이너 이미지 다운로드
2. 컨테이너 생성 및 라이브러리 다운로드
3. 서비스의 코드 실행을 위한 런타임 준비
4. 서비스 제공

컨테이너를 실행할 때, 1 ~ 3 번 과정이 포함되어 시작하면 콜드 스타트(Cold Start), 4 번 과정에서 시작되면 웜 스타트(Warm Start)라 한다. 서버리스 플랫폼인 Knative 의 경우 컨테이너 이미지 다운로드 과정은 가장 초기 과정인 서비스 배포에만 이루어지므로 본 논문에서는 컨테이너 생성이 되는 2 번 과정부터를 콜드 스타트라 칭한다.

3. 연속된 콜드 스타트

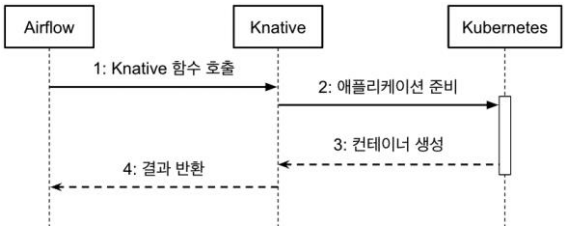
A. 워크플로우

본 논문에서는 연속된 콜드 스타트를 유발하는 환경을 만들기 위해 워크플로우 관리 도구인 Apache Airflow 를 사용했다. Airflow 를 사용하면 Python 프로그래밍 언어로 DAG(Directed Acyclic Graph) 형태의 워크플로우를 만들 수 있다.[5] 연속된 콜드 스타트 환경을 재현하기 위해 [그림 1]과 같이 워크플로우를 작성하였다.



[그림 1] 3 개의 연속된 Task 를 가진 워크플로우

각 Task 에는 Knative 서비스를 호출하는 Python 코드가 작성되어 있으며 그 과정은 [그림 2]와 같다. 본 논문은 서비스 유형과 무관하게 실행 시간만을 고려하기 때문에 특별한 작업을 수행하지 않는 서비스를 대상으로 했다. Task 1 부터 3 은 각각 3 초, 7 초, 5 초 동안 프로세스를 멈추는 작업을 하는 서비스를 호출한다.



[그림 2] Airflow 에서 호출되는 Knative 콜드 스타트 과정

B. 문제점

[그림 1]과 같이 서버리스 플랫폼에 배포된 서비스를 차례대로 호출하는 경우, 각 서비스 컨테이너의 생성 과정을 기다려야 한다. 로컬 환경에서 Knative 를 구축하여 콜드 스타트와 웜 스타트의 시간 차이가 약 1 초가 측정되었고, 워크플로우 전체에 대해서는 약 3 초의 시간 차이가 발생하였다.

4. Pre-Warming 아키텍처

A. Knative 서비스의 Warm-up Handler

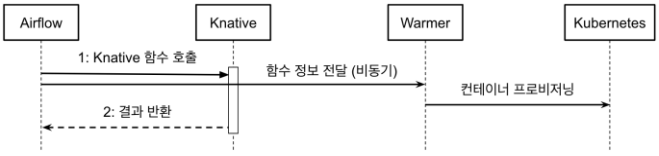
Knative 서비스는 HTTP 요청을 받아 처리하는 웹 서버 형태를 갖는다. 본 논문에서는 Knative 서비스를 웜 스타트 상태로 만들기 위해 추가적인 엔드포인트를 작성했다. Knative 는 콜드 스타트 상태에서 서비스로 들어가는 요청이 있으면 컨테이너를 무조건 생성하기 때문에, [그림 3]과 같이 /warmup 경로에 연결되어 특별한 작업을 하지 않는 핸들러를 추가했다.

```
package main
…
   func main() {
      http.HandleFunc("/", main_handler)
      http.HandleFunc(“/ping”, warmup_handler)
   …
}
```

[그림 3] Go 언어로 작성된 Knative 서비스 예제

B. Pre-Warmer 컴포넌트

본 논문에서 연속된 콜드 스타트 완화를 위해 의존성 있는 서비스들의 컨테이너를 미리 생성하는 Pre-Warmer 컴포넌트를 구현했다. Pre-Warmer 는 실행된 워크플로우의 정보를 입력으로 받고, 첫 번째 서비스로부터 의존성 있는 서비스의 Warm-up Handler 를 호출하여 컨테이너를 미리 생성한다. 컨테이너가 미리 생성된다면 [그림 4]처럼 결과를 Knative 로부터 바로 받을 수 있기 때문에 콜드 스타트로 인한 지연시간을 줄일 수 있다.



[그림 3] Airflow 에서 호출되는 Knative 웜 스타트 과정

[그림 2]와 같이 모든 함수에 대해 컨테이너를 생성할 때에는 전체 시간이 약 18 초 소요되었지만, 의존성이 있는 다음 함수의 컨테이너를 미리 생성함으로써 Task 2 와 Task 3 작업의 지연시간 약 2 초를 줄일 수 있다.

5. 결론 및 향후 계획

서버리스 컴퓨팅은 컨테이너 기반 애플리케이션을 배포 및 관리하는 새로운 컴퓨팅 패러다임이다. 본 논문에서는 서버리스 컴퓨팅의 문제점인 콜드 스타트가 실제 사용될 수 있는 워크플로우에서는 연속된 콜드 스타트로 인해 전체 실행시간에 많은 영향을 줄 수 있음을 보였다. 또한, 그에 대한 해결책으로 의존성이 있는 서버리스 서비스의 경우 미리 실행 환경을 준비하는 방법을 사용하여 연속된 콜드 스타트를 완화하는 아키텍처를 제안하였으며, Apache Airflow 와 Knative 를 사용하여 컨테이너 기반 서버리스 플랫폼에서 전체 실행시간이 줄어드는 것을 확인했다.

워크플로우의 형태는 시퀀스 형태 뿐만 아니라 더 복잡한 형태가 될 수 있다. 그에 따른 Pre-Warming 과정에서 비효율적인 리소스가 발생할 수 있다. 향후 연구에서는 Apache Airflow 를 사용한 다양한 워크플로우를 대상으로 효율적인 Pre-Warming 기법을 연구할 예정이다.

감사의 글

본 연구는 산림청(한국임업진흥원) 산림과학기술 연구개발사업 ‘2022427C10-2224-0801’의 지원에 의하여 이루어진 것입니다.

참고문헌

[1] Hossein Shafiei, Ahmad Khonsari, Payam Mousavi, “Serverless Computing: A Survey of Opportunities, Challenges, and Applications”, ACM Computing Surveys, 2022

[2] Simon Eismann, Joel Scheuner, Erwin van Eyk, Maximilian Schwinger, Johannes Grohmann, Nikolas Herbst, Cristina L. Abad, Alexandru losup, “A Review of Serverless Use Cases and their Characteristics”, https://arxiv.org/abs/2008.11110, 2021

[3] Kubernetes, https://kubernetes.io/

[4] Knative, https://knative.dev/docs/

[5] Airflow, https://airflow.apache.org/