서버리스 컴퓨팅 환경에서의 캐싱 기법 동향에 관한 연구

김도균, 김동현, 정현서, 김경연, 박상오*
중앙대학교, 중앙대학교, 중앙대학교, 중앙대학교, *중앙대학교

dgkim@cslab.cau.ac.kr, dhkim@cslab.cau.ac.kr,
hsjung@cslab.cau.ac.kr, kykim@cslab.cau.ac.kr, *sopark@cau.ac.kr

A Study on the Trend of Caching Techniques in a Serverless Computing Environment

Kim Do Gyun, Kim Dong Hyeon, Jung Hyun Seo, Kim Kyeong Yeon, Park Sang Oh*
Chungang Univ., Chungang Univ., Chungang Univ., *Chungang Univ.

요 약

서버리스 컴퓨팅은 클라우드 자원의 효율적인 관리와 비용 절감을 가능하게 하지만, 초기화 지연 시간인 콜드 스타트(Cold Start) 문제는 여전히 주요한 과제로 남아있다. 이를 해결하기 위해 다양한 캐싱 기법이 연구되어 왔으며, 최근에는 단순한 컨테이너 유지를 넘어 부분 캐싱(Partial Caching) 및 공유(Sharing) 기법으로 진화하고 있다. 또한, 딥러닝 추론과 같은 고성능 워크로드가 서버리스로 이동함에 따라, 거대 모델 로딩 시간을 단축하기 위한 캐싱 전략도 등장하고 있다. 본 논문에서는 서버리스 환경의 전통적인 캐싱 기법들을 분류하고, FaaSnap 과 RainbowCake 를 중심으로 최적화 기법을 분석한다. 나아가 FaSei 를 통해 AI 추론을 위한 최신 캐싱 연구 동향을 고찰한다.

Ⅰ. 서 론

서버리스 컴퓨팅(Serverless Computing)은 개발자가 물리적 인프라나 운영체제를 직접 관리할 필요 없이, 애플리케이션 로직에만 집중할 수 있게 하는 클라우드 실행 모델이다. 특히 이벤트 기반으로 코드를 실행하는 FaaS(Function-as-a-Service)는 요청량에 따라 인스턴스를 0 개까지 줄이거나(Scale-to-zero) 무한히 확장할 수 있는 유연성과, 실제 실행 시간만큼만 비용을 지불하는 경제성을 바탕으로 현대 클라우드 애플리케이션의 핵심 패러다임으로 자리 잡았다. 그러나 서버리스 플랫폼은 요청이 없을 때 자원을 회수하므로, 새로운 요청이 들어왔을 때 실행 환경(샌드박스, 런타임 등)을 처음부터 구성해야 하는 콜드 스타트 문제가 발생한다. 콜드 스타트는 일반적으로 수백 밀리 초에서 수 초가 소요되며[1], 짧은 함수 실행 시간 대비 큰 비율의 지연을 차지해 사용자 체감 성능을 악화시킨다.

이 문제를 해결하기 위해 처음에는 한 번 초기화된 실행 환경을 일정 시간 동안 메모리에 상주시켜 재사용하는 '웜 스타트(Warm Start)' 방식을 표준적으로 사용해 왔다. 그러나 이 방식은 메모리 효율성과 지연 시간 단축이라는 상충되는 목표 사이에서 딜레마를 겪는다. 모든 함수에 대해 실행 환경을 미리 준비해 두는, 전체 캐싱(Full Caching)은 콜드 스타트를 제거할 수 있는 가장 확실한 방법이지만, 산발적으로 호출되는 함수들을 위해 수백 MB 의 메모리를 점유하고 있는 것은 막대한 자원 낭비를 초래한다.

따라서 최근의 연구들은 단순한 웜 스타트를 넘어, 디스크 스냅샷을 활용해 메모리 점유 없이 복원 속도를 높이거나, 실행 환경의 구조를 계층화하여 메모리를 절약하는 고도화된 캐싱 기법으로 진화하고 있다. 또한, 최근 서버리스 컴퓨팅을 활용한 AI 모델 서빙과 관련한 최적화도 시도되고 있다. 본 논문에서는 이러한 최신 서버리스 캐싱 기술의 발전 양상을 분석한다.

Ⅱ. 실행 환경 캐싱 및 최적화 기법의 발전

전통적인 전체 캐싱은 가장 효과적인 콜드 스타트 완화 방법이지만, 유휴 상태의 실행 환경이 메모리를 지속적으로 점유하여 자원 낭비를 초래한다는 한계가 존재한다. 이를 극복하기 위해 하나는 디스크 기반의 스냅샷을 빠르게 복원하여 메모리 상주 없이도 웜 스타트와 유사한 속도를 내는 것이고, 다른 하나는 메모리에 상주하되 전체가 아닌 세분화하여 캐싱하는 방식으로 연구가 진행되고 있다.

1) 스냅샷 기반 복원 가속화

메모리 상주 비용을 줄이기 위한 대안으로, VM 이나 마이크로 VM(예: Firecracker[2])의 상태를 디스크에 스냅샷으로 저장해두고 필요시 복원하는 기술이 주목받고 있다. 하지만 기존 스냅샷 기술은 게스트 메모리를 필요할 때 읽어오는 Lazy Loading 방식을 사용하더라도 디스크 I/O 로 인한 성능 저하가 발생한다.

따라서 FaaSnap[3]은 이러한 I/O 병목을 해결하기 위해 제안된 마이크로 VM 스냅샷 최적화 플랫폼이다. FaaSnap 은 스냅샷 복원 시 필요한 메모리 페이지 집합(Working Set)을 사전에 파악하여 병렬적으로 프리페치(Concurrent Paging)함으로써, 실행 중 발생하는 페이지 폴트 대기 시간을 획기적으로 줄인다. 또한 게스트 메모리 영역을 그 내용과 특성에 따라 구분하는 Per-Region Memory Mapping 을 적용하여,

불필요한 I/O 작업을 제거하고 복원 속도를 웜 스타트에 근접한 수준으로 끌어올린다. 이는 실행 환경을 메모리에 계속 유지하지 않고도 콜드 스타트 오버헤드를 최소화하려는 접근 방식이다.

2) 계층별 캐싱 및 공유 방식

스냅샷 방식과 달리, 메모리에 실행 환경을 유지하되 그 비용을 낮추는 접근법도 존재한다. 기존의 전체 컨테이너 캐싱은 컨테이너가 특정 함수에 종속되어 재사용성이 낮다는 문제가 있다. RainbowCake[4]는 이를 해결하기 위해 부분 캐싱(Partial Caching)과 공유(Sharing)를 결합하여 메모리 효율성을 극대화한다.

RainbowCake 는 컨테이너 초기화 과정을 Bare(인프라), Lang(언어 런타임), User(사용자 코드)의 세 레이어로 분리하여 관리한다. 이 기법은 유휴 상태의 컨테이너를 즉시 종료하거나 전체를 유지하는 대신, 상위 레이어인 User 레이어만 제거하고 하위 레이어인 Lang 또는 Bare 상태로 캐싱하는 전략을 취한다. 이렇게 확보된 레이어 컨테이너는 동일한 언어 런타임을 사용하는 다른 함수들과 공유될 수 있다. 이는 메모리 낭비를 최소화하면서도, 인프라 및 런타임 초기화 단계를 건너뛰어 콜드 스타트를 효과적으로 완화하는 방식이다.

Ⅲ. AI 추론을 위한 모델 중심 캐싱 전략

최근 서버리스 환경에서 딥러닝(DL) 모델 추론을 수행하려는 시도가 늘어나고 있다. 하지만 DL 모델은 거대한 크기로 인해 로딩 시간이 전체 콜드 스타트 시간의 약 83% 이상을 차지할 정도로 병목 현상이 심각하다[5]. 앞서 언급한 RainbowCake 와 같은 기법은 런타임 초기화 시간을 줄여주지만, 실행 직전에 수백 MB 이상의 모델을 로딩해야 하는 문제는 여전히 해결하지 못한다. 또한, 엣지 서버와 같이 자원이 제약된 환경에서는 거대한 모델을 포함한 전체 실행 환경을 메모리에 캐싱하는 것이 불가능에 가깝다. 따라서 AI 추론을 위해서는 인프라 레벨을 넘어 모델 데이터 자체의 로딩을 최적화하는 새로운 접근이 필요하다.

이러한 문제를 해결하기 위해 FaSei[5]는 AI 모델의 실행 특성에 맞춘 새로운 캐싱 전략을 제시한다. FaSei 는 딥러닝 추론이 레이어 순차적으로 진행된다는 점에 착안하여, 전체 모델을 로딩하지 않고 초기 레이어만 로딩된 상태에서 추론을 시작하는 Model Lazy Loading 기법을 도입한다. 하지만 단순히 Lazy Loading 기법만 사용할 경우 로딩 속도가 추론 속도를 따라가지 못해 추론 속도가 중단되는 Inference Bubble 이 발생할 수 있다. 이를 방지하기 위해 로딩 시간이 오래 걸리거나 병목을 유발할 수 있는 특정 레이어들을 선별하여 미리 캐싱(Layer-wise Caching)하고, 나머지는 Lazy Loading 으로 처리하는 전략을 취한다. 이 방식은 각 모델 레이어의 파라미터 크기와 연산량의 이질성을 고려하여 최적의 캐싱 대상을 선정함으로써 전체 응답 시간을 단축한다. 이는 실행 환경 중심의 최적화를 넘어, 애플리케이션 내부의 모델 구조를 고려한 캐싱으로 연구 흐름이 진화하고 있음을 시사한다.

Ⅳ. 결론

서버리스 컴퓨팅의 콜드 스타트 문제 해결을 위한 연구는 단순한 메모리 상주 전략에서 벗어나 다각화되고 있다. FaaSnap 은 마이크로 VM 의 스냅샷 복원 과정을 최적화하여 메모리 점유 없이도 빠른 시작을 가능하게 했으며, RainbowCake 는 컨테이너 구조를 계층화하여 메모리 효율성과 재사용성을 극대화했다. 더 나아가 FaSei 는 AI 워크로드의 특성을 반영하여 모델 레이어 단위의 정교한 캐싱 전략을 제시했다. 향후 서버리스 캐싱 기술은 실행 환경의 경량화와 거대 언어 모델의 추론 워크로드 특성을 결합한 통합적인 최적화 방향으로 발전할 것으로 전망된다.

ACKNOWLEDGMENT

이 논문은 정부(과학기술정보통신부)의 재원으로

한국연구재단의 지원(RS-2024-00345869)과

정부(과학기술정보통신부)의 재원으로 한국연구재단 지원(RS-2025-02217071)을 받아 수행된 연구임.

참 고 문 헌

[1] Mohammad Shahrad, Rodrigo Fonseca, Inigo Goiri, Gohar Chaudhry, Paul Batum, Jason Cooke, Eduardo Laureano, Colby Tresness, Mark Russinovich, & Ricardo Bianchini (2020). Serverless in the Wild: Characterizing and Optimizing the Serverless Workload at a Large Cloud Provider. In *2020 USENIX Annual Technical Conference (USENIX ATC 20)* (pp. 205- 218). USENIX Association.

[2] Alexandru Agache, Marc Brooker, Alexandra Iordache, Anthony Liguori, Rolf Neugebauer, Phil Piwonka, & Diana-Maria Popa (2020). Firecracker: Lightweight Virtualization for Serverless Applications. In 17th USENIX Symposium on Networked Systems Design and Implementation (NSDI 20) (pp. 419- 434). USENIX Association.

[3] Ao, L., Porter, G., & Voelker, G. (2022). FaaSnap: FaaS made fast using snapshot-based VMs. In Proceedings of the Seventeenth European Conference on Computer Systems (pp. 730- 746). Association for Computing Machinery.

[4] Yu, H., Basu Roy, R., Fontenot, C., Tiwari, D., Li, J., Zhang, H., Wang, H., & Park, S.J. (2024). RainbowCake: Mitigating Cold-starts in Serverless with Layer-wise Container Caching and Sharing. In Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 1 (pp. 335- 350). Association for Computing Machinery.

[5] Huang, Z., Dong, F., Guo, X., & Yin, D. (2025). FaSei: Fast Serverless Edge Inference with Synergistic Lazy Loading and Layer-wise Caching. In IEEE INFOCOM 2025 - IEEE Conference on Computer Communications (pp. 1-10).