Flame: A Centralized Cache Controller for Serverless Computing

Yanan Yang[1], Laiping Zhao[1], Yiming Li[1], Shihao Wu[1], Yuechan Hao[2], Yuchi Ma[2], Keqiu Li[1]
TANKLAB, CIC, Tianjin University, China[1], Huawei Cloud[2]

Introduction

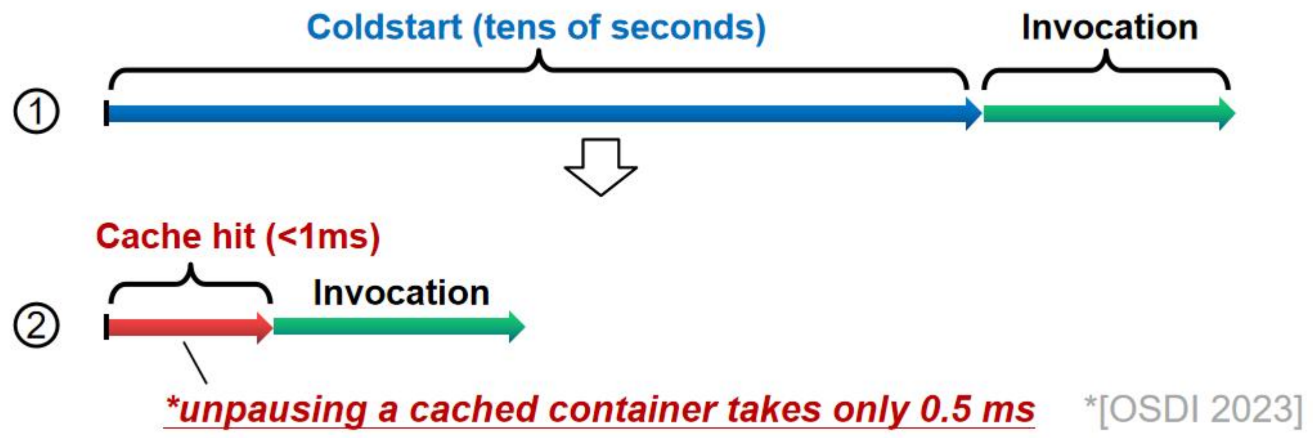
Caching function is a promising way to mitigate coldstart overhead in serverless computing. However, as caching also increases the resource cost significantly, how to make caching decisions is still challenging. We find that the prior “local cache control" designs are insufficient to achieve high cache efficiency due to the workload skewness across servers.

In this paper, inspired by the idea of software defined network management, we propose Flame, an efficient cache system to manage cached functions with a “centralized cache control" design. By decoupling the cache control plane from local servers and setting up a separate centralized controller, Flame is able to make caching decisions considering a global view of cluster status, enabling the optimized cache-hit ratio and resource efficiency. We evaluate Flame with real-world workloads and the evaluation results show that it can reduce the cache resource usage by 36% on average while improving the coldstart ratio by nearly 7× than the state-of-the-art method.

Keywords: Serverless Computing, Keep-alive, Hotspot Function, Coldstart

Motivation

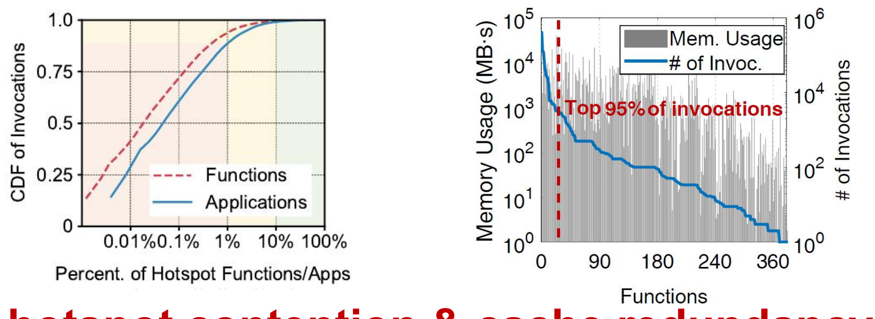
- **Function caching is a potential way to eliminate coldstart**
- **However, it also brings Non-trival cloud cost**



**unpausing a cached container takes only 0.5 ms* *[OSDI 2023]

- **Q: How to improve the function cache efficiency?**

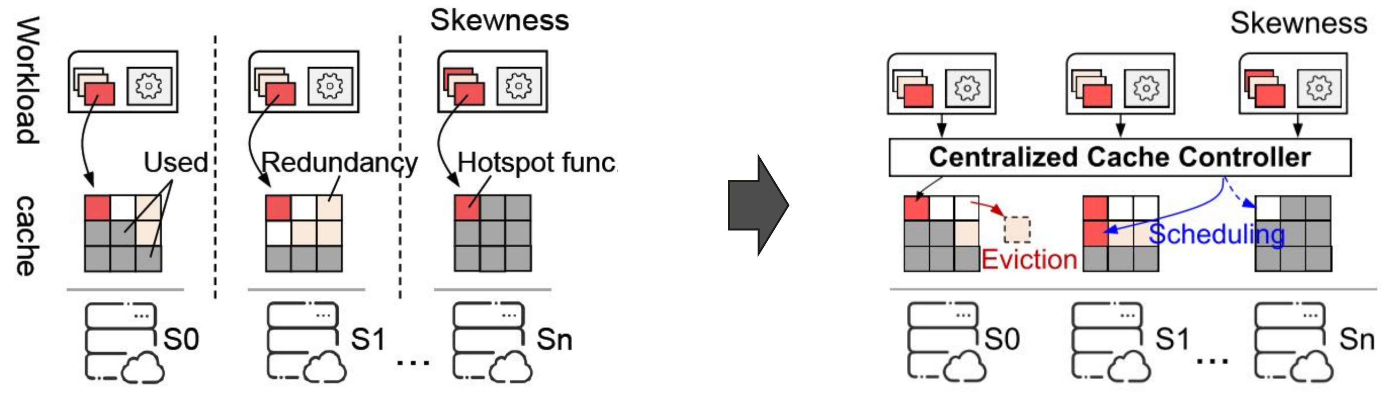
Existing Approaches:
Local cache control: e.g., [AWS’16], [ATC’20], [ASPLOS’21]
Shortages:
Low cache efficiency under workload skewness



hotspot contention & cache redundancy

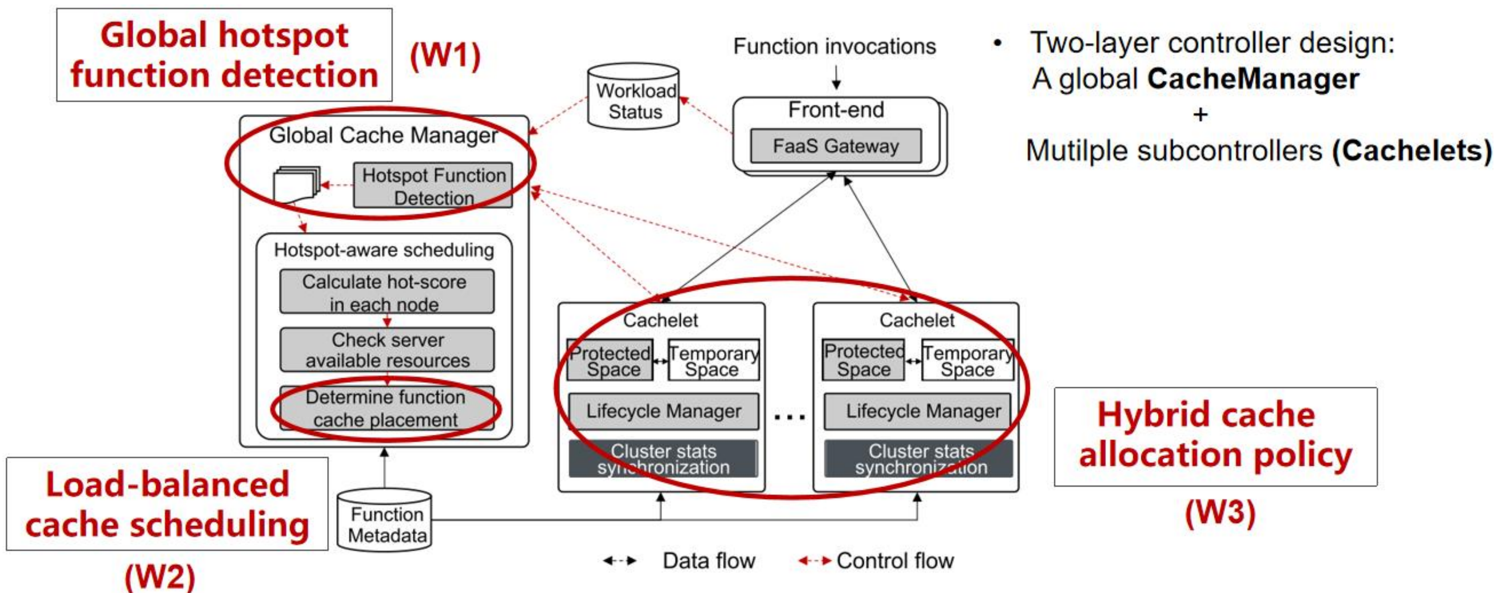
Methods & System design

- **Key idea: Using a centralized cache control system to efficiently manage the cached functions via a global view of cluster status**



(a) Local Cache Control (b) Centralized Cache Control

- **The 3W challenges:**
- Which function should be cached? (W1)
- Where the cached functions should be cached? (W2)
- When the cached functions should be released? (W3)

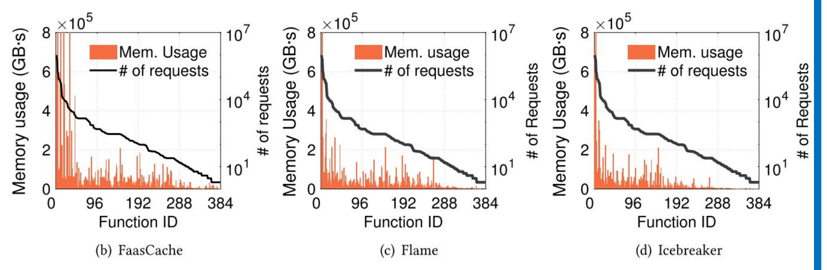

Global hotspot function detection
(W1)
Function invocations
Workload Status
Front-end
FaaS Gateway
Global Cache Manager
Hotspot Function Detection
Hotspot-aware scheduling
Calculate hot-score in each node
Check server available resources
Determine function cache placement
Cachelet
Protected Space
Temporary Space
Lifecycle Manager
Cluster stats synchronization
Cachelet
Protected Space
Temporary Space
Lifecycle Manager
Cluster stats synchronization
Hybrid cache allocation policy
(W3)
Load-balanced cache scheduling
(W2)
Function Metadata
Data flow
Control flow

- Two-layer controller design:
A global **CacheManager**
+
Mutilple subcontrollers **(Cachelets)**

An overview of Flame

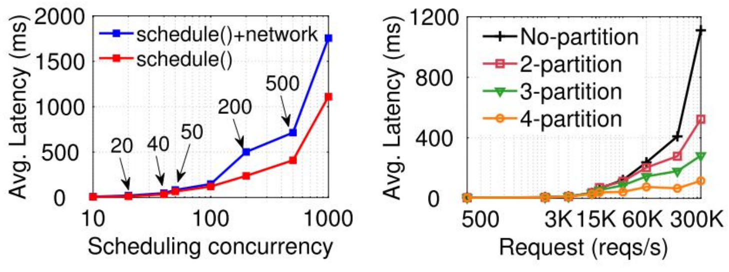
Evaluation

- Flame can reduce the cache resource usage by 26%-54% on average and reduce the coldstart ratio by more than 7×



(b) FaasCache (c) Flame (d) Icebreaker

- Flame generates negligible overhead in system resource usage and decision latency, and it can be easily extended in large scale of workload scenarios



(a) Cache scheduling time (b) Scheduling time optimization

Contact

Email: ynyang@tju.edu.cn
Github:http://github.com/Flame/ca-cheResearch/

[Bib] Yanan Yang, Laiping Zhao, Yiming Li, etc. Flame: A Centralized Cache Controller for Serverless Computing. ASPLOS (4) 2023: 153-168.