Optimizing Resource Utilization with Jaigu

A review by:
Jackson A. Goldberg

UNIVERSITY of WASHINGTON

1

Table of Contents

Slides:

2

Problems with AWS and Azure

- At present AWS Lambda and Azure Functions allocate too many resources.
- AWS and Azure do not adapt efficiently to changes in workload
- Current Mindset: better to be prepared and use more resources to ensure good performance.

3

Effects of this Mindset

- Increased utilization of resources means the developers are charged more for resources that aren't being used.
- Lack of good adaptability to traffic means that during high usage users suffer poor performance, and during low traffic developers suffer additional costs.

4

Benefits of Solving this Problem

- Money saved for developer
  - Less resources used
- More consistent performance for the user
  - More resources during high traffic

5

Related Works

- Machine learning research
  - Kumari et al. (2022) - Directed Acyclic Graph
  - Mampage et al. (2023) - deep reinforcement learning
- Decoupled Resource Allocation
  - Bilal et al. (2021) - memory and cpu decoupling
- Delayed decision making
  - Sinha et al. (2024) - postponing resource decisions

6

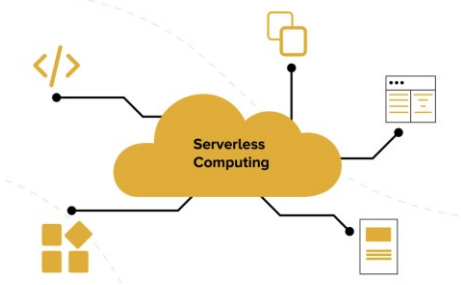
Important Ideas

- Cloud service providers provide a good default option for serverless functions but individual systems can be improved for efficiency.
  - "Guessing" upcoming traffic
  - Independently provisioning resources

7

Jaigu

- Traffic Prediction
- Cheat Sheet Creation
  - Capacity Table
- Dual-Staged Scaling



8

Traffic Prediction

- Decoupling prediction and scheduling
  - Pre-decision Scheduling
  - maximum number of instances without violating Quality of Service (QoS)
- Random Forest Regressor
  - Tested on data from: Huawei Cloud

9

Capacity Table

- Make quick decisions
  - Preloads table with instructions for various workloads
  - Created using ML
  - Significantly reduces decision making
- Maximum Capacity of each Function
  - Stores max capacity for each function without violating QOS

10

Dual-Staged Scaling

- "Pauses" unused instances
  - Essentially caching them
  - If unused for a significant amount of time they are deleted
- Solves cold start problem
  - Cached instances are started much faster than completely new instances.

11

Key contributions

- Pre-decision scheduling
  - Decoupling prediction and scheduling
- Dual-Staged Scaling
  - Release
  - Eviction
- Capacity Tables

12

Experimental Evaluation pt. 1

- Hardware Setup
  - 24 machines
    - Intel Xeon E5-2650 CPU (48 logical cores)
    - 128GB memory
  - One machine dedicated to control plane
  - Other machines are worker nodes

13

Experimental Evaluation pt. 2

- Baseline Systems
  - Kubernetes
  - Gsight
  - OWL
- Functions
  - Model inference
  - Batch processing applications (img resize, linpack)
  - Log processing
  - File processing (gzip)

14

Experimental Evaluation pt. 3

- Scheduling Costs
  - Reduced by 81%-93% compared to Gsight
- Cold Start Latency
  - Lowered by 57%-69% using container fork
- Function Density
  - 54.8% higher than Kubernetes

15

Author's Conclusions

- Main Points
  - Pre-decision Scheduling
  - Dual-Staged Scaling
  - Performance Gains
  - Adaptability
  - Practicality
- Future Directions
  - More seamless integration

16

Critique: Strength

- Performance improvements
  - Faster with less resources
  - More practical for the intended usage
- Solves a important problem
  - Saves developer money
  - Maintains QOS

17

Critique: Weakness

- Complexity of implementation
  - Requires data for ML task
  - Capacity table reliance requires set up
- Edge cases
  - Caching instances in edge cases cause more resource utilization

18

Critique: Evaluation

- Strong statistics measured
  - Hits at the most important stats for serverless computing
- Well documented process
  - Easy to replicated if desired
- Well explained
  - Easy for reader to understand why specific tests were chosen

19

Gaps

- Still new
  - While tested this paper has much to gain from more tests
- More deployments of different sizes
  - Testing on different machines with different hardware
- Comparisons over time
  - Showing how much this could save with a real world implementation

20

Questions??

21