Intelligent Serverless Computing Systems

The talk was about improving serverless computing by efficiently reconfiguring available resources. Dr. Wang presented his paper titled "*Accelerating Serverless Computing by Harvesting Idle Resources*". Serverless computing systems have changed the workflow of many applications providing easy and fast deployments. In his paper, he tackles the issue of static resource provisioning of cloud functions to improve resource utilization without depending on users' function configuration.

The authors propose a new serverless resource manager (RM) called *Freyr* which estimates the CPU and memory saturation points for functions, identifies over provisioned and under provisioned functions, and harvests the wasted resources to support under provisioned functions. *Freyr* sits in the controller of a serverless computing framework and interacts with the container system (e.g., Docker) that executes the function invocations. *Freyr* is designed to be event driven with multi process support so that the arrival of a function triggers resource harvesting decisions.

Freyr contains a deep reinforcement learning agent that estimates the resource demands for the functions. It collects information such as number of invocations remaining, available CPU cores, and available memory for the platform and average CPU peak, average memory peak, average inter arrival time, average execution time and baseline execution time for the functions. Using this information, a score network is trained to score and justify each allocation option for a function invocation.

To avoid potential performance degradation of functions from reprovisioning, they implement a safeguard mechanism. Functions with resource usage below 80% of the user requested level are considered to be over provisioned. For such overprovisioned functions, *Freyr* checks if the usage peak of last invocation is 80% of the allocation and triggers the safeguard mechanism.

The authors deployed and evaluated *Freyr* on an OpenWhisk cluster using realistic workloads from public serverless benchmarks and invocation traces sampled from Azure Functions traces. They then compare it with three baseline resource managers (RM) – Fixed RM, Greedy RM and ENSURE. They use slowdown value for each of these models to measure the performance of a function invocation. Lower slowdowns mean lower function response latency and are preferred. Evaluation results show that *Freyr* outperformed other baseline RMs. *Freyr* achieved lowest average slowdown of 0.82 whereas Fixed RM, Greedy RM and ENSURE had 1.0, 1.12, and 1.78 respectively. Freyr provided 18% faster function executions and 32.1% lower response latency for the testing workload compared to default RM in OpenWhisk. And it harvested idle resources from 38.8% of function invocations accelerating 39.2% of invocations.

Comments/Thoughts

- Well written paper. Nice flow and easy to read.
- Was comparison with additional resource managers possible?
- Could there be a better way of selecting a function from a pool of over provisioned functions for reconfiguration?
- What if the user has purposefully allocated low compute resources to avoid the cost of high resource utilization? Maybe some sort of mechanism similar to safeguard mechanism could be enforced.
- Unexpected network traffic, say from DDOS attacks, could lead to the allocation of a lot of resources resulting in huge unwanted costs.
- Since their scoring mechanism is dependent on a neural network, it takes time to learn its features when deployed for the first time. Some sort of transfer learning could be helpful to transfer the already learnt features from the previously deployed agent.