ECE750-T37 Assignment #2 Report G13

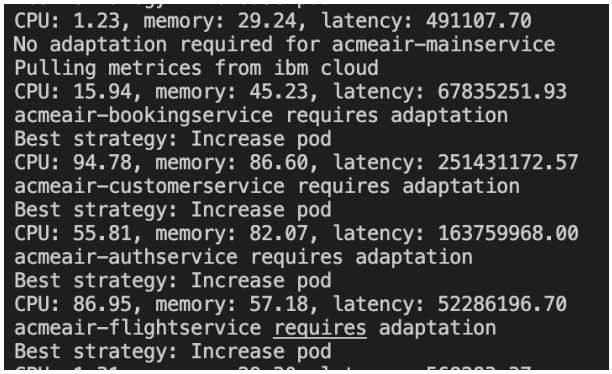
1. **Design of driver program**

   When the monitoring phase is completed, an automated data analysis process is initiated. Our analyzer reads the generated JSON files and transforms the data into DataFrames for further analysis. Subsequently, we compute the average values for specific properties of interest, such as CPU usage, memory usage, and maximum latency. These average values are then compared to predefined quality requirements (QRs) to determine the need for adaptation. If any of the QRs fail to meet the criteria, it triggers the adaptation process. Our feedback loop offers three potential configurations:

   - Add 500m core CPU resources.
   - Add 512MB to the existing memory.
   - Create a new pod with identical configuration.

   For each of these configurations, we calculate a utility function to identify the most optimal strategy, and the results are displayed in the console.

   Figure 2 shows the example output of our analyzer component.



We utilized two key metrics, **cpu_quota_used_percent** to monitor CPU usage and **memory_limit_used_percent** to track memory usage. In addition, we closely monitored maximum response times and request durations. Our feedback loop, integrated at the gateway, extensively analyzed these metrics and harnessed them for well-informed decision-making. Our primary focus has been on achieving **self-optimization**. Our system is in a constant pursuit of opportunities to reduce maximum latency while adhering to essential quality objectives, such as efficient CPU and memory resource utilization. Within our utility function, we assigned significant weight to latency, allocating the remaining components to other relevant attributes. This strategy ensures our system maintains low latency while consistently meeting the full spectrum of quality requirements.

Additional work completed includes:

- Developing shell scripts to trigger JMeter to continuously send requests to our website with varying thresholds every 5 minutes. Creating a Docker container for JMeter and deploying it on OpenShift to ensure a constant flow of requests.
- Undertaking code refactoring to enhance scalability and maintainability.

2. **Implementation of analysis approach**

   Main source of uncertainties

   From our perspective, fluctuating traffic represents the primary source of uncertainties. A surge in the volume of requests and active users per hour can significantly elevate the level of

uncertainty within the server environment. This influx may trigger issues such as congestion, delays, errors, or system failures. The server's capacity may prove insufficient to efficiently handle the increased demands, resulting in suboptimal performance and reduced availability. Particularly, varying workloads can lead to a notable difference in CPU utilization, as illustrated in Figure 1, further amplifying uncertainty. To mitigate these uncertainties stemming from fluctuating traffic, the server should possess the capability to autonomously adjust its capacity and resource allocation in response to periods of high traffic load.

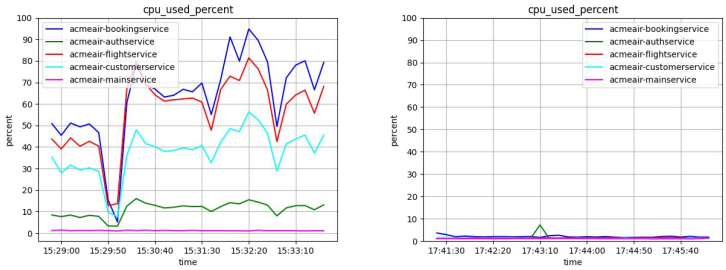


Fig. 1. CPU used percentage of heavy traffic and light traffic

Quality requirements of Acme Air

After thorough analysis of the matrix, we have established the following quality requirements:

- CPU usage percentage per 5-minute period should not exceed 50%.
- Memory usage percentage per 5-minute period should not exceed 50%.
- Response latency per 5-minute period should not exceed 2e7.

These requirements are essential to maintain the Error Rate and Quality of Service for the Acme Air server, as exceeding these thresholds would significantly compromise its performance.

Utility related to adaptation goals

we employed a Rule-Based Approach for systematic analysis. We've experimented with various configurations to gauge their impact on latency, allowing us to subsequently employ our utility function to select the optimal configuration.

Here's a set of potential configurations for the 'acmeair-bookingservice'.

Configuration	CPU	Memory	Pod	CPU Usage	Memory Usage	Latency (1e08M)
1	0.5	512	1	95%	82%	5.320747
2	1	512	1	62%	79%	2.830374
3	0.5	1024	1	83%	65%	3.137130
4	0.5	512	2	53%	58%	1.918004

CPU utility preference

utility = 1 if CPU Usage $\leqslant$ 25%, 0.5 if > 25% and $\leqslant$ 50%, 0 if > 50%

Memory utility preference

utility = 1 if Memory Usage Percentage $\leqslant$ 25%, 0.5 if > 25% and $\leqslant$ 50%, 0 if > 50%

Latency utility preference

utility = 1 if Respond Latency Times $\leqslant$ 1e7, 0.5 if > 1e7 and $\leqslant$ 2e7, 0 if > 2e7

Resource Cost utility preference (CPU, Memory, Pod)

utility = 1 if cost $\leqslant$ 0.5, 0.5 if > 0.5 and $\leqslant$ 1, 0 if > 1

Expected Utilities:

$$U_c = W_{cpu} * P_{cpu} + W_{memory} * P_{memory} + W_{latency} * P_{latency} + W_{cost} * P_{cost}$$