**Assignment 2**

**1. Summary**

In this assignment, we used the number of threads as the primary source of uncertainty for Acme Air, as the number of threads represents the effective number of requests. We believe that what Acme Air needs to ensure first is that the number of times the pods crash and restart does not exceed once every 5 minutes, and the request failure errors due to system performance issues do not exceed 5%. For this purpose, we will use a self-optimizing mode to allow the system to optimize itself, ensuring the system can meet the targets by increasing and decreasing CPU and memory. We will attempt to describe the relationship between CPU, memory, and thread count by running Acme Air under different load conditions. We will try to build a utility function and display the analysis approach.

**2. Program Architecture & Design**

The program is divided into three functions, which can be enabled or disabled by commenting out the function calls located in the main function.

The first function, getData(), is used to collect the metrics that should be gathered during the operation of Acme Air. This part mainly uses the code from assignment 1, but some content has been added. The input of the code is the monitoring duration length, and the output is a jsonl dataset recording various related metrics and several pictures depicting the trend of metrics. We used the try-except method to prevent program crashes caused by pods crashing and restarting.

The second function, analyze(), is used to analyze the correlation between various metrics and the number of threads. Based on the data collected by getData(), it will use linear regression to analyze the correlation of various metrics to the thread pool service by service and will draw the corresponding scatter plots and regression functions. Eventually, it will output specific regression functions for each metric and several intuitive pictures showing their relevance.

The third function, selfOptimize(), is used for adaptively adjusting the size of CPU and memory. It will check whether it is necessary to increase or decrease CPU and memory usage through the utility function. When the CPU usage is below 20%, it will reduce CPU usage; when the CPU usage is above 80%, it will increase CPU usage. When the memory usage is below 20%, it will reduce memory usage; when the memory usage is above 80%, it will increase memory usage. By making adjustments in the above manner, it will effectively reduce the risk of pods crashing and restarting and reduce request failure errors due to performance problems. Also, during system idle times, it will automatically lower the system performance used, thereby reducing the overall performance requirements of the system. Once this function is called, it will run permanently. When a request fails due to issues like pod crashes and restarts, it will print out the error. When the function operates normally, it will summarize Acme Air every 30 seconds, adjust the CPU and memory usage service by service, and print out the system status and adjustment reports.

**2. Utility Function**

**3. Analysis Approach**

The detailed steps are shown in the figure below. We adopted the ECA (Event Condition Action) approach. Every five minutes, the program will collect data. After the collection is completed, it reaches the analysis event. It will face two conditions: the percentage of CPU usage and the percentage of memory usage. Different actions will be taken based on different situations. If the CPU% is higher than 80%, it will increase the CPU allocation; if it's less than 20%, it will decrease the CPU allocation. If the memory% is higher than 80%, it will increase the memory allocation; if it's less than 20%, it will reduce the memory allocation. After the action is completed, it will enter the next cycle.

A diagram of a company

Description automatically generated

**Thank you.** *Group 11*

*All the plots and raw data along with the scripts used have been provided separately. The codebase used has been updated to* [*GitHub repository*](https://github.com/Nikunj47/Acmeair) *and is made up to date. In case of any discrepancy, kindly reach out at n2chhabr@uwaterloo.ca.*