# Micro-Service System Troubleshooting

Henry Zheng Sahand Sabour Samuel Pegg

**Abstract**

With the rapid growth of large micro-service based systems, detecting anomalies in system operations has become a significantly essential task. In addition, the root causes of such anomalies must also be detected to prevent major costs and losses. Due to the large number of existing calls and complex relationships between different micro-services in modern systems, manual inspection of such micro-services has become extremely challenging and rather impossible. Hence, developing a set of methods that accomplish these tasks within a timely manner and with high accuracy is highly necessary. In this report, we propose an online algorithm that analyzes data generated from a micro-service based system and detects the occurring anomalies and their corresponding root causes.

Keywords: micro-service, anomaly detection, root cause analysis, root cause detection

## 1 Introduction

With the recent trends, increasing number of modern day large-scale systems are utilizing micro-services in their system architecture design. In a micro-service based architecture, the system operations are provided as different services, which are loosely coupled (i.e. independent from each other), organized, and highly maintainable, analyzable, and testable. Hence, this type of architecture allows large-scale and complex systems to provide faster and more reliable services to their users. However, a major disadvantage of these systems is the massive number of calls between different services and their complex relationships; which in turn makes finding faulty services and errors rather challenging.

In the case that a faulty micro-service causes an error in the system, this occurrence would be referred to as an anomaly, which should be detected as soon as possible. Accordingly, the anomaly should be traced back to the faulty service that caused it. This process is referred to as troubleshooting and by accomplishing this task, the system maintainers would be able to fix the faulty service and any subsequent problems before experiencing major costs and losses. Therefore, implementing an algorithm that detects system's anomalous behavior and is able to both accurately and efficiently identify its root cause is essential.

In this project, we were tasked to design an online algorithm to perform these tasks on an incrementally published dataset (i.e. real-time data) generated by a micro-service based system through Kafka producer. More specifically, our task was to analyze the provided time series data, detect anomalous behavior, discover anomalous system nodes at the time of this occurrence, and find the Key Performance Indicator (KPI) that is the root cause of this anomaly. The rest of this report is organized as follows: in section 2, we summarize the relative literature for these tasks and acknowledge the work of top teams for this competition; in section 3, we further explain the problem statement and provide our implemented algorithms for this project; section 4 includes a discussion of our results as well as the lessons learned; lastly, we conclude the report in section 5.

## 2 Related Work

Hello

## 3 Methodology

In this section, we describe the problem statement and our implemented algorithms.

### 3.1 Problem Statement

In this project, we were provided with three KPI data sources: ESB, Trace, and Host data.

**ESB business indicator (ESB):** it is provided every minute and mainly demonstrates the number of requests for the *osb_001* service and the overall success rate of these request during each minute. Once an anomaly occurs, assuming at a point t in time, the success rate is expected to be lower than 1. Accordingly, t would be recorded to be used for further analysis in the other two KPI data sources.

**Trace:** it is provided for every request and consists of several micro-service calls, referred to as spans. This section of the data demonstrates the start and elapsed time for each span, the databases that were accessed (if any), the trace of the span and its host. Upon finding anomalous time t in the ESB data, the time around t is to be investigated in order to detect anomalous spans (i.e. nodes with unusually long response time) and ultimately, realize faulty service nodes.

**Host:** are provided in the (timestamp, value) format and includes the host service name and the called operation. Upon finding the faulty service nodes, this data can be explored to find anomalous values for a KPI, which would then be flagged as a root cause.

### 3.2 Proposed Method

Similar to the previous section, our proposed method consists of three parts corresponding to the three different KPI data sources that we are provided: ESB, Trace, and Host analysis.

**ESB business indicator (ESB):**

**Trace:**

**Host:**

## References